Open Advanced Process Control

Benutzerhandbuch

Version 4.0

(c) 2008-2015 by OpenAPC Project Group

Table Of Contents

Table of Contents

1 Copyright....3
2 Rechtliches....4
3 Sicherheit....5
4 Überblick....6
4.1 Die Softwarestruktur....6
5 ControlRoom....7
5.1 Systemarchitektur....7
5.2 Projekte Erstellen und Ausführen....11
5.3 Verwendung des OpenEditor....11
5.3.1 Der HMI-Editor des OpenEditors....12
5.3.2 Der Flow-Editor des OpenEditors....12
5.3.2.1 Flow-Objekte des Floweditors....14
5.3.2.1.1 Stand-Alone Flow-Objekte....14
5.3.2.1.1.1 Externe Laser-Flow-Objekte....14
5.3.2.1.1.2 Externe Motion-Flow-Objekte....19
5.3.3 Die Liste der Plugged Devices im OpenEditor....22
5.4 Benutzung des OpenDebuggers....22
5.5 Benutzung des OpenPlayer....23
5.5.1 OpenPlayer Kommandozeilenparameter....23
5.6 Benutzung des OpenHPlayer....24
5.7 Benutzung des Interlock Servers....24
5.7.1 Interlock Server Modi....25
5.7.1.1 Einfacher Lokaler Modus....25
5.7.1.2 Gespiegelter Lokaler Modus....25
5.7.1.3 Einfacher Modus über Netzwerk....26
5.7.1.4 Redundanter Modus über Netzwerk....26
5.8 Benutzung des OpenPlugger....27
6 CNConstruct....28
6.1 CNConstruct modifizieren....28
7 BeamConstruct....30
7.1 Sicherheitshinweise....30
7.2 Überblick....30
7.3 Position innerhalb des Gesamtsystems....30
7.4 Schnellstartanleitung für BeamConstruct....32
7.5 Projektkonfiguration....35
7.6 Stifteinstellungen....36
7.6.1 Allgemeine Stifteinstellungen....37
7.6.1.1 Stiftparameter-Wizard....37
7.7 IOSelect-Modus....38
7.8 BeamConstruct modifizieren....38
7.9 BeamLock....40
7.10 BeamServer Remote Control Interface....41
7.10.1 Überblick....41
7.10.2 Verwendung....41
7.10.3 BeamServer starten....42
7.10.4 Remote Control Kommandos....43
8 CorrCorrect....48
8.1 Overview....48
8.2 Benutzung....48
8.3 Korrekturdefinitionsdialog....49
8.4 Korrektur einzelner und mehrere Punkte....49
8.5 Dialog zur räumlichen Korrektur....50

1 Copyright

Dieses Dokument ist © by OpenAPC Project Group. OpenAPC, ControlRoom, CNConstruct und BeamConstruct sind registrierte Handelsmarken.

Andere hier beschriebene Software, die nicht zum OpenAPC Softwarepaket gehört, ist © und TM / Handelsmarke ihrer jeweiligen Eigentümer.

Fedora, RedHat, RedHat Enterprise Linux, RHEL sind Copyright / Handelsmarken von Red Hat Inc.

Linux ist Copyright / Handelsmarke von Linus Torvalds.

Ubuntu ist Copyright / Handelsmarke von Canonical.

Microsoft, Windows, das Windows-Logo und andere sind Copyright / Handelsmarken der Microsoft Corporation.

Panasonic und Minas sind Copyright / Handelsmarken von Matsushita.

MDrive und MDrive+ sind Copyright / Handelsmarken von Schneider Electronic.

MySQL ist Copyright / Handelsmarke von Oracle.

Isel und Isel Wafer Handler Robot sind Copyright / Handelsmarken von Isel.

JoyWarrior ist Copyright / Handelsmarke der Code Mercenaries Hard- und Software GmbH.

SCAPS, SAM, CCI, FEB and USC sind Copyright / Handelsmarken der SCAPS GmbH.

RTC, RTC3, RTC4, RTC5, RTCscanalone, SCANLAB und andere sind Copyright / Handelsmarken der SCANLAB AG.

SiRF ist Copyright / Handelsmarke von SiRF Technology Inc.

Weecoboard, Weecoboard 4M und Aptasys sind Copyright / Handelsmarken von Aptasys s.r.l.

SP-ICE2, Raylase und andere sind Copyright / Handelsmarken der Raylase AG.

ETH6608, Sintec und andere sind Copyright / Handelsmarken von Sintec Optronics.

PSC, PSC140P, 3rdEye und andere sind Copyright / Handelsmarken von 3rdEye.

Coherent, Avia und andere sind Copyright / Handelsmarken von Coherent Inc.

Sill und andere sind Copyright / Handelsmarken der Sill Optics GmbH & Co. KG.

Alle anderen in diesem Dokument erwähnten Handelsmarken sind Handelsmarken oder registrierte Handelsmarken ihrer jeweiligen Eigentümer.

Alle Rechte, die hier nicht explizit gewährt werden, bleiben vorbehalten.

2 Rechtliches

Die Eignung der Software ist durch den Benutzer festzustellen, es werden keinerlei Garantien oder Zusagen über die Tauglichkeit, Fehlerfreiheit oder Vollständigkeit der Software oder dieses Dokuments gemacht. Die Benutzung der hier beschriebenen Software sowie die Verwendung der Informationen aus diesem Dokument geschieht vollständig auf eigene Gefahr. Das gilt insbesondere auch dann, wenn Fehler, Auslassungen, unvollständige oder falsche Informationen in der Software oder in diesem Dokument bereits bekannt sind.

Fehler, Irrtümer, Änderungen, Auslassungen in diesem Dokument bleiben vorbehalten. Die Sicherheit der Funktion von Anlagen, Maschinen und anderem sind in jedem Fall durch geeignete Hardwaremaßnahmen sicherzustellen, die hier beschriebene Software ist nicht in der Lage, dies zu gewährleisten.

Die Lizenzbedingungen im englischen Handbuch gelten ergänzend. Sollten einzelne Bestimmungen dieser rechtlichen Hinweise oder der englischen Lizenzbedingungen unwirksam oder undurchführbar sein oder unwirksam oder undurchführbar werden, bleibt davon die Wirksamkeit der rechtlichen Hinweise und der englischen Lizenzbedingungen im Übrigen unberührt. An die Stelle der unwirksamen oder undurchführbaren Bestimmung soll diejenige wirksame und durchführbare Regelung treten, deren Wirkungen der wirtschaftlichen Zielsetzung am nächsten kommt, welche mit der unwirksamen bzw. undurchführbaren Bestimmung durch den Hersteller verfolgt wurde. Die vorstehenden Bestimmungen gelten entsprechend für den Fall, dass sich die rechtlichen Hinweise oder die englischen Lizenzbedingungen als lückenhaft erweisen.

3 Sicherheit

Die Komponenten des OpenAPC Softwarepakets sind erstellt worden, um verschiedene Maschinen und Geräte anzusteuern. Diese Maschinen und Geräte können unter Umständen die Gesundheit von Personen beeinträchtigen und/oder anderweitig Schaden anrichten. Vor der Installation der Software und vor deren Benutzung ist zu überprüfen, ob alle relevanten Sicherheitsmaßnahmen ergriffen wurden und ob alle hardwaregestützten Schutzmechanismen aktiv und wirksam sind. Der Benutzer ist während der ganzen Zeit dafür verantwortlich, dass die Software in einer Umgebung betrieben wird, welche allen relevanten Sicherheitsregeln entspricht, so dass auch durch Fehlbedienungen oder Fehlfunktionen kein Schaden angerichtet werden kann.

4 Überblick

Dieses Dokument beschreibt das Open Advanced Process Control (OpenAPC) Softwarepaket mit den darin enthaltenen Funktionen, Optionen und Softwarekomponenten. Abhängig von der tatsächlich genutzten Softwarevariante können einige der hier beschriebenen Funktionen fehlen, nicht benutzbar sein oder nur in einer begrenzten Variante verfügbar sein.

Das Open Advanced Process Control Softwarepaket ist für verschiedene Hard- und Softwareplattformen verfügbar. Abhängig von den Eigenheiten der verwendeten Betriebssysteme kann der genaue Speicherort von Programmen, Plug-Ins oder anderen Daten abweichen. Innerhalb dieser Dokumentation wird immer vom gewöhnlich zu erwartenden Speicherort von solchen Dateien ausgegangen. Dieses Dokument enthält keine betriebssystemspezifischen Informationen, hier soll auf die jeweiligen Dokumentationen der Betriebssysteme verwiesen werden.

4.1 Die Softwarestruktur

Das OpenAPC-Paket ist eine Sammlung von Softwarekomponenten und zusätzlichen Tools welche für unterschiedliche Zwecke eingesetzt werden können. Ihre Funktionalitäten können abhängig vom Einsatzzweck auf verschiedenen Wegen kombiniert werden. Das OpenAPC-Softwarepaket enthält die folgenden Hauptkomponenten, welche sich wiederum in mehrere Unterapplikationen aufteilen können:

- **ControlRoom** ist eine Visualisierungs- und Prozesssteuerungssoftware welche unter anderem dazu verwendet werden kann, HMIs zu erstellen, Maschinen anzusteuern, Produktionsprozesse zu kontrollieren oder auch um verschiedene Aufgaben in der Heimautomatisierung zu übernehmen. Die ControlRoom-Software besteht aus separaten Programmteilen, welche zur Erstellung und Entwicklung (OpenEditor und OpenDebugger) und zum Ausführen einer HMI/Prozesssteuerung verwendet werden können (OpenPlayer/OpenHPlayer, OpenPlugger, InterlockServer, ...)
- **CNConstruct** ist eine CAD-Applikation, welche verwendet werden kann, um generische CNC- und Materialverarbeitungsabläufe auf einfache Art in einer komfortablen WYSIWYG-Umgebung zu erstellen. Mit CNConstruct erstellte Projekte können zur deren Abarbeitung nahtlos in ein ControlRoom Prozessteuerungsprojekt integriert werden.
- **BeamConstruct** ist eine CAD-artige Applikation ähnlich CNConstruct, welche auf laserbasierte Prozesse und Lasermarkieroperationen (Schweißen, Schneiden, Gravieren, Reinigung von Oberflächen, Lasersintern, ...) spezialisiert ist. Solche Lasermarkierprojekte können inklusive der optional enthaltenen Ansteuerung von externen Geräten entweder direkt aus BeamConstruct heraus oder aber mittels eines ControlRoom-Projektes, welches die BeamConstruct-Funktionalitäten in eine HMI-Umgebung integriert, ausgeführt werden.

5 ControlRoom

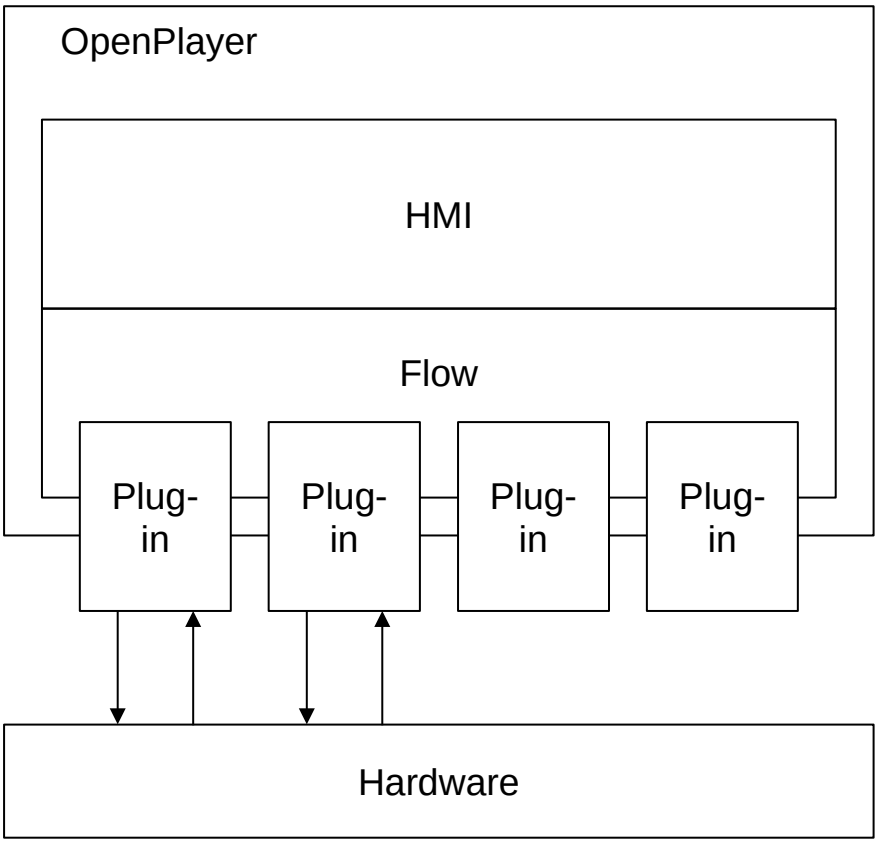
Die Komponente “ControlRoom” des OpenAPC Softwarepaketes kann verwendet werden, um HMIs (Human Machine Interfaces, zu Deutsch auch MMI, Mensch-Maschine-Interfaces) zu erstellen und um Prozessabläufe zu definieren. Diese Komponente setzt sich aus folgenden Programmen zusammen:

- der Editor (bezeichnet als OpenEditor, nicht auf allen Zielsystemen verfügbar), mit dem neue Projekte erstellt und bestehende verändert werden können
- der Debugger (benannt als OpenDebugger, nicht auf allen Zielsystemen verfügbar), mit dem ein existierendes, mit dem OpenEditor erstelltes Projekt ausgeführt werden kann; dieser Debugger bietet die Möglichkeit, während der Ausführung den Datenfluss zu analysieren und die Funktion eines ControlRoom-Projektes zu verifizieren
- der Player (benannt als OpenPlayer), mit dem Projekte auf dem Zielsystem ausgeführt werden können; im Unterschied zum OpenDebugger besitzt dieser keinerlei Funktionen zur Fehleranalyse sondern ist darauf ausgerichtet, ein Projekt unter Verwendung von möglichst wenig Ressourcen möglichst effizient auszuführen
- die spezielle Playervariante OpenHPlayer, welche Projekte ohne jeglichen HMI-Anteil, also ohne grafische Anzeige auszuführen, von diesem werden nur die definierten Programmabläufe (so genannte “Flows”) ohne jegliche grafische Anzeige ausgeführt
- der optionale Interlock Server (benannt als OpenIServer) welcher im Hintergrund ausgeführt werden kann, um Zustände und Werte eines laufenden Projektes zu verwalten und dessen Datenflüsse und Abläufe zu beeinflussen
- die optionale Laufzeitumgebung für Plug-Ins (der OpenPlugger), welche verwendet werden kann, um Plug-Ins mit Hardwarezugriffsfunktionen außerhalb des OpenPlayers laufen zu lassen; diese würden dann über dem Interlock Server mit anderen Komponenten der Laufzeitumgebung kommunizieren

Daneben existieren noch verschiedene Plug-Ins, welches jedes für sich eine eigene, abgegrenzte Spezialaufgabe übernimmt und damit die Hauptapplikation um zusätzliche Funktionalitäten wie z.B. Hardwarezugriffe erweitert. Solche Plug-Ins sind externe Programme, welche technisch gesehen eine DLL/eine Shared Library sind, die während der Programmlaufzeit dynamisch geladen werden. Auf dem Zielsystem ist es deswegen möglich, alle unbenutzten Plug-Ins aus der Installation zu löschen um somit Speicherplatz zu sparen.
ACHTUNG: wenn dabei versehentlich Plug-Ins gelöscht werden, die innerhalb eines ControlRoom-Projektes verwendet werden, so kann dieses Projekt nicht mehr ordnungsgemäß funktionieren!

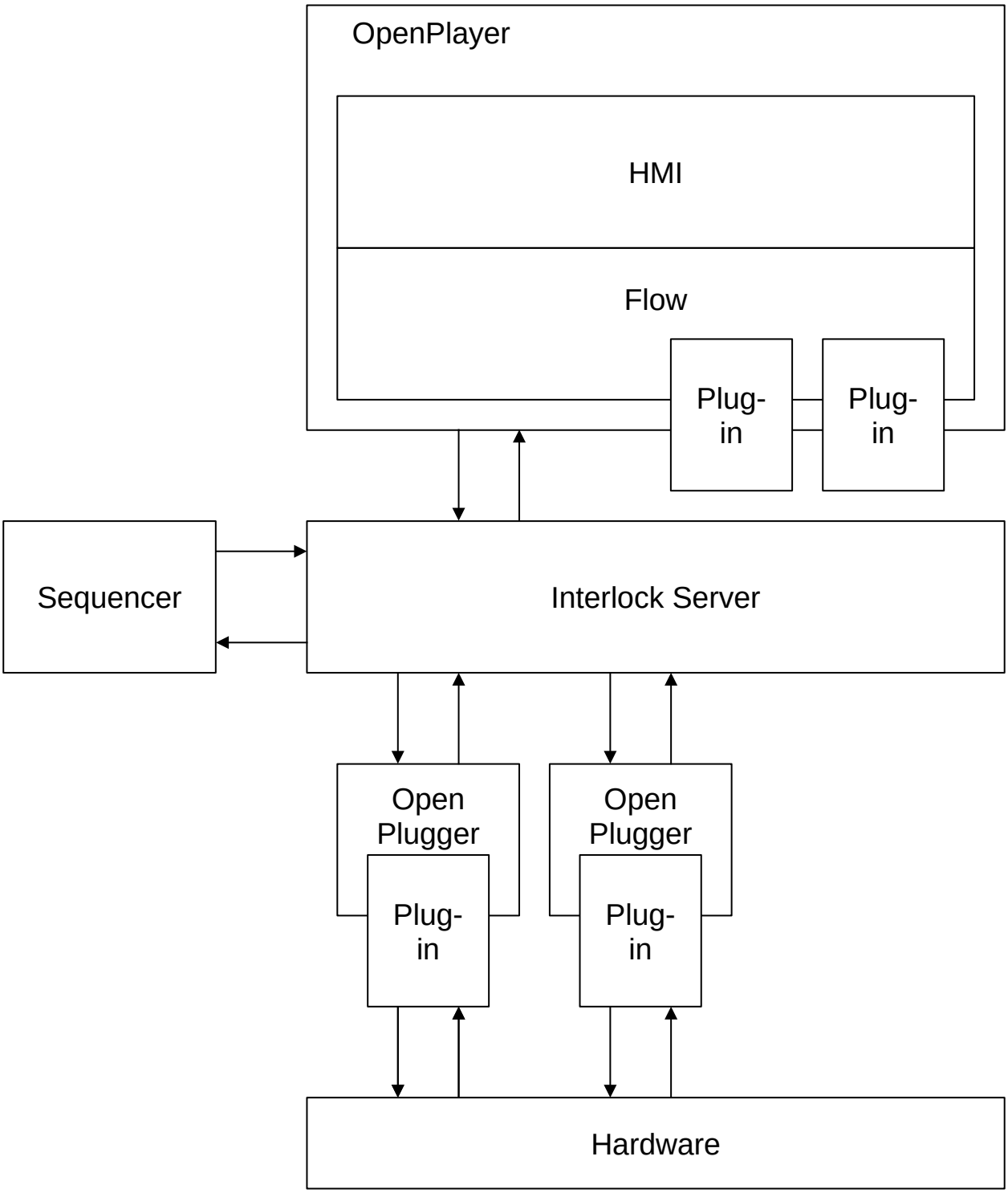
5.1 Systemarchitektur

Die Softwarekomponente “ControlRoom” bietet verschiedene Möglichkeiten, eine Prozessteuerungsumgebung zu erstellen und zu verwenden. Welche Variante die geeignetste ist, hängt von der Aufgabenstellung und den Anforderungen an das zu steuernde System ab.



Die einfachste Architekturvariante besteht ausschließlich aus dem OpenPlayer. Hier werden die komplette Visualisierung und der logische Ablauf vom OpenPlayer übernommen. Ebenso die Plug-Ins, diese sind mit dem Player verbunden und kommunizieren direkt mit diesem und mit den intern über den so genannten Flow (die logischen Verknüpfungen, welche den Programmablauf festlegen) verbundenen anderen Elementen. So ein Projekt kann schnell und leicht erstellt werden, hat jedoch einige gravierende Nachteile, welche beim Design beachtet werden sollten: wachsen die Anforderungen und damit die in ein solches Projekt integrierten Funktionalitäten, so kann dieses schnell kompliziert, unübersichtlich und schwer wartbar werden, da keine klare, sauber getrennte Struktur existiert, in welcher funktional ähnliche Teile in eigenen Schichten angeordnet sind. Aus diesem Grund wird empfohlen, diese Struktur nur für einfache, kleine Projekte zu verwenden, bei denen keine Gefahr besteht, dass diese ausufern und im Laufe der Zeit stark wachsen.

Für komplexere Umgebungen ist es empfehlenswert, die gesamte Struktur zu entzerren und jede Komponente in eine eigene Schicht zu verlagern, die zu ihrer Aufgabenstellung passt. D.h. alle Plug-Ins, welche externe Hardware ansteuern, sollten von der HMI und dem OpenPlayer entkoppelt werden. Das erfordert eine zusätzliche Kommunikationsinstanz, welche die dann getrennten Elemente wieder miteinander verbindet: den Interlock Server. Dieser befindet sich dann zwischen Player und Plug-Ins – welche wiederum über je eine Instanz des OpenPlugger an den Interlock Server angebunden werden:



Diese Softwarestruktur bietet die Möglichkeit, weitere Softwarekomponenten an den Interlock Server anzudocken, mit diesen alle Komponenten der Umgebung zu beeinflussen und damit auch den Ablauf zu steuern. Diese Zusatzkomponenten (im Bild exemplarisch durch den Sequencer dargestellt) können Hardwarezustände überwachen (gemeldet durch die jeweiligen OpenPlugger und die diesen zugeordneten Plug-Ins), können auf Benutzereingaben reagieren (gemeldet durch den OpenPlayer) und können den Ablauf steuern und beeinflussen (durch Setzen/Ändern der im Interlock Server gespeicherten Daten und Zustände). Dieser Sequenzer kann eine eigenes Programm sein, ein LUA- oder Instruction List (IL) Skript (IL entspricht einer Anweisungsliste / AWL) oder eine weitere Instanz eines OpenPlayer auf einem anderen System. Da sämtliche Kommunikation mit dem Interlock Server über TCP/IP abgewickelt wird, können diese Komponenten zur Lastverteilung auch über ein Netzwerk verteilt sein.

Die Vorteile einer solchen Struktur sind:

- Sie lässt sich leicht und einfach erweitern, in dem neue Elemente hinzugefügt werden, die sich über die bestehende Kommunikationsinfrastruktur des Interlock Server anbinden
- Die Struktur ist klar geordnet, die Kommunikationswege sind einfach und eindeutig (es wird alles über den Interlock Server geroutet, es existieren keine verwirrenden Kreuz- und Querverbindungen zwischen all den Elementen, die miteinander kommunizieren müssen, so entstehen auch keine Laufzeitprobleme, weil Informationen auf Grund eines unerwarteten Kommunikationsweges doppelt oder zu spät ankommen)
- Der Flow im OpenPlayer kontrolliert nur noch die GUI-Elemente, es existiert keine Vermischung mit dem Flow der Ablaufsteuerung, die Gesamtstruktur im OpenPlayer wird einfacher
- Die Struktur ist deutlich näher am allgemein anerkannten OSI Schichtenmodell (https://de.wikipedia.org/wiki/OSI-Modell)
- Da alle Elemente via TCP/IP miteinander kommunizieren, können alle logischen Teile (“Sequencer”, “OpenPlayer”, “Interlock Server”, “OpenPlugger”) optional auf je einem eigenen Computersystem betrieben werden, so dass es möglich ist, komplexe Systeme über ein Netzwerk verteilt aufzubauen

5.2 Projekte Erstellen und Ausführen

Die Prozedur zur Erstellung und Verwendung eines Projektes besteht aus nur einigen einfachen Schritten. Startpunkt für alles ist der OpenEditor. Hier kann ein neues Projekt erstellt oder ein bestehendes zur Erweiterung seiner Funktionalität geladen werden. Innerhalb des OpenEditor wird das grafische Benutzerinterface im HMI-Editor erstellt. Die logischen Beziehungen und Datenflüsse zwischen diesen werden anschließend im Flow-Editor erstellt. Für den Fall, dass die erweiterte Softwarestruktur mit externen, über OpenPlugger verwaltete Plug-Ins verwendet werden soll, kommt die dritte Komponente zum Einsatz, die sich hinter der Tab-Pane verbirgt „Plugged Devices“.

Im nächsten Schritt kann dieses Projekt im OpenDebugger ausgeführt werden. Hier wird eine ähnliche Umgebung wie auf dem Zielsystem geboten, in der es zusätzlich aber möglich ist, den tatsächlichen Programmablauf zu analysieren, schrittweise abzuarbeiten und so die Funktionalität des Projektes zu überprüfen.

Wenn das Projekt wunschgemäß arbeitet, kann es mit dem OpenPlayer auf dem Zielsystem ausgeführt und verwendet werden.

5.3 Verwendung des OpenEditor

Das Fenster des OpenEditor besteht aus drei Hauptbestandteilen, welche über Tab-Panes erreicht werden können:

- Der HMI-Editor, in welchem das grafische Benutzerinterface erstellt werden kann
- Der Flow-Editor, in welchem die logischen Verbindungen zwischen Elementen des Benutzerinterfaces und den Plug-Ins festgelegt werden können; dieser bietet die Möglichkeit den Programmablauf grafisch zu definieren und die logischen Verbindungen zwischen einzelnen HMI-Elementen und Plug-Ins durch Einzeichnen von Flows festzulegen
- Die Plugged Devices Liste, in welcher all diejenigen Plug-Ins festgelegt und konfiguriert werden können, die später nicht innerhalb des Players sondern extern in je einer eigenen Instanz des OpenPlugger laufen sollen; diese Liste ist nur dann zugänglich, wenn in den Projekteinstellungen die Verwendung des Interlock Server konfiguriert wurde

Zwischen den Editoren und der Liste kann über die Tabs (“Karteikartenreiter”) am oberen Fensterrand hin und her geschaltet werden. Oberhalb dieser Tabs finden sich zusätzlich verschiedene Toolbarbuttons, wovon einige zu identischen Menüpunkten passen. Diese können für verschiedene Aktionen verwendet werden:

- Erzeugen eines neuen Projektes
- Laden eines existierenden Projektes
- Speichern des aktuellen Projektes mit dem bestehenden oder unter einem neuen Namen
- Ändern oder Löschen von HMI- oder Flow-Elementen
- Untersuchen des Projektes auf Fehler und mögliche Probleme
- Debuggen und Testen des aktuellen Projektes im OpenDebugger
- Ausführen des Projektes im OpenPlayer

Diese Toolbarbuttons sind für alle Programmteile aktiv, sowohl für HMI- und Flow-Editor als auch für die Deviceliste. Das gilt ebenso für die Menüs, hier finden sich folgende zusätzliche Funktionen:

- Ändern der Projekteinstellungen
- Suchen eines Elementes anhand seines Namens

5.3.1 Der HMI-Editor des OpenEditors

Die HMI-Editor Tab-Pane bietet die Möglichkeit, ein grafisches Benutzerinterface zu erstellen, welches dem Endanwender später zur Bedienung der Steuerung gezeigt wird. Die Größe dieses Interfaces wird dabei in den globalen Projekteinstellungen festgelegt. Diese sollte normalerweise der Bildschirmauflösung des gewählten Zielsystems entsprechen, um den gesamten Anzeigebereich auszufüllen.

Ein neues GUI-Element kann durch einen Rechtsklick mit der Maus im HMI-Panel angelegt werden. Es öffnet sich ein Kontextmenü, welches verschiedene Elemente wie Container, Buttons, Eingabefelder, Anzeigeelemente, Bilder und anderes anbietet. Sollen mehrere Elemente des gleichen Typs hintereinander erzeugt werden, so kann durch wiederholtes Betätigen der Taste „F2“ ein neues Element des zuvor verwendeten Typs erneut erzeugt werden. Dieses neue Element wird an der Position des letzten Mausklicks (linke Maustaste) positioniert.

Auf der rechten Seite des Editors existiert eine Foldbar, welche die gleichen GUI-Elemente wie das Kontextmenü anbietet. Diese kann ebenfalls verwendet werden, um neue Elemente zu erzeugen. So lange wie ein Eintrag in dieser Foldbar selektiert ist, erzeugt jeder langsame Klick mit der linken Maustaste ein neues Element dieses Typs an der angeklickten Position.

Bereits existierende GUI-Elemente können selektiert werden, um diese zu verändern oder die Eigenschaften festzulegen. Um diese zu selektieren, müssen sie bei heruntergedrückter linker Maustaste umrahmt werden, hilfsweise wird dabei ein Rechteck gezeichnet, welches diese Elemente umfasst. Einige Elemente können auch durch einen direkten Linksklick mit der Maus selektiert werden, das funktioniert – abhängig von dessen Typ – aber nicht mit allen GUI-Elementen.

Wenn ein oder mehrere Elemente selektiert sind, so können diese editiert und modifiziert werden. Im Kontextmenü (rechte Maustaste) sind dazu jetzt mehrere neue Menüpunkte aktiv und benutzbar. Mit diesen Funktionen des Kontextmenüs können diese Elemente dupliziert oder gelöscht und ihre Eigenschaften bearbeitet werden.
Eine spezielle Funktion die an dieser Stelle ausgeführt werden kann, ist die Ausschneiden/Einfügen Operation. Wird ein HMI-Element ausgeschnitten, so wird es exklusiv in einem internen Puffer gehalten. Die zu diesem Element möglicherweise zugehörigen Flow-Verbindungen bleiben bestehen. Sobald dieses Element an anderer Position eingefügt wird, so wird der interne Puffer geleert, es ist also nicht möglich, dieses Element mehrfach einzufügen und somit zu kopieren. Als Ergebnis dieser Operation wird das HMI-Element an seiner neuen Position sichtbar und die zugehörigen Flow-Verbindungen bleiben in Funktion. Wird die Funktion „Ausschneiden“ zweimal aufgerufen, ohne das zuerst ausgeschnittene Element zuvor mit „Einfügen“ neu zu platzieren, so wird das zuerst ausgeschnittene Element durch diesen zweiten Aufruf gelöscht.

Ist ein einzelnes GUI-Element selektiert, so wird das durch einen blauen Rand um dieses Element herum symbolisiert. Dieser Rahmen ist multifunktionell: in den Ecken und an den Seiten befinden sich graue Quadrate, mit denen man durch Ziehen die Größe verändern kann. Die untere rechte Ecke bietet dabei eine Skalierfunktion, bei der die Proportionen des Elementes erhalten bleiben.

Die Bereiche zwischen diesen Quadraten können verwendet werden, um die Position des Elementes durch Ziehen mit der linken Maustaste zu verändern..

Wenn mehrere Elemente selektiert sind, so ist die Skalierfunktion nicht verfügbar, in diesem Fall können diese GUI-Elemente nur gemeinsam in der Position verändert werden.

5.3.2 Der Flow-Editor des OpenEditors

Im Flow-Editor ist es möglich, die grafischen Elemente des Benutzerinterface (welche zuvor im HMI-Editor erstellt, positioniert und konfiguriert wurden) logisch miteinander zu verknüpfen, so dass diese

- Sich gegenseitig ansteuern können
- Andere Flow-Elemente ansteuern und beeinflussen können
- Daten zu angeschlossenen Geräten senden und diese damit steuern können

Für jedes zuvor erstellte HMI-Element existiert ein Flow-Objekt, dessen angebotene Funktionalität mit dem Zweck des HMI-Elementes korrespondiert. Zusätzlich sind im Flow-Editor weitere Objekte verfügbar, die nicht direkt zu einem Element des Benutzerinterface gehören, welche aber verwendet werden können, um Abläufe zu steuern, Daten zu manipulieren oder auf externe Hardware zuzugreifen.

Innerhalb des Flow-Editors können dort angeordnete Flow-Objekte miteinander verbunden werden. Jede Verbindung entspricht dabei einer Art Datenleitung, über die während der Ausführung Informationen von einem Flow-Objekt zu einem anderen übertragen werden können. So eine Datenleitung sieht zwar wie eine elektrische Verbindung aus, verhält sich aber anders als von elektrischen Geräten gewohnt: während eine tatsächliche elektrische Leitung einen Zustand normalerweise für eine definierte Zeit hält, ist der Zustand eines Datenflusses in einer solchen Verbindung nur kurzzeitig, es wird jeweils immer nur ein abgeschlossenes Datenpaket übertragen. Verglichen mit elektrischen Verbindungen verhält sich so eine Daten- oder Flow-Verbindung folgendermaßen:

Elektrisches Signal	**Verhalten einer Flow-Verbindung**
Konstantes elektrisches Signal: eine Leitung hat einen definierten, festen Pegel (z.B. HIGH oder LOW), es kann nicht zwischen mehreren Signalen unterschieden werden, die den gleichen Pegel haben, da auf der Leitung keine Änderung stattfindet	Konstanter logischer Status: wenn ein Flow-Objekt ein logisches Signal an einem Ausgang abgibt (z.B. ein digitales LOW oder HIGH), wird dieses entlang aller Flow-Verbindungen an diesem Ausgang zu allen verbundenen Eingängen übertragen. Nachdem alle Eingänge dieses Signal erhalten haben, hat die Flow-Verbindung selber keinen Zustand mehr. Wird nun erneut ein logisches Signal angegeben, so erfolgt erneut eine Übertragung zu allen angeschlossenen Flow-Objekten – und das auch dann, wenn die übertragene Information identisch ist und sich zum vorherigen Signal nicht verändert hat.
Elektrischer Impuls: eine Leitung wird geschaltet ((z.B. auf HIGH oder LOW) und nach einer definierten Zeit wieder zurückgeschaltet (z.B. auf LOW oder HIGH)	Übertragung eines Impulses: ein solcher Puls besteht immer aus separaten Übertragungen zweier Zustände. Zuerst wird der Startwert übertragen (z.B. ein digitales HIGH oder LOW), anschließend wird der Ursprungswert übertragen (z.B. ein digitales LOW oder HIGH) um auf diesen zurückzuschalten. Zwischen diesen Übertragungen hat die Flow-Verbindung selber keinen definierten Zustand.

Zusammenfassend kann gesagt werden, dass die Übertragung von Informationen über die Flow-Verbindungen nicht mit elektrischen Signalen verglichen werden kann, sondern vielmehr mit der Übermittlung einzelner Informationspakete. Jedes Flow-Objekt speichert dabei die empfangenen Daten so lange, bis ein neues Datenpaket empfangen wird.

Flow-Objekte unterstützten an ihren Ein- und Ausgängen verschiedene Datentypen. Ein Ausgang eins Flow-Objektes kann dabei nur mit einem Eingang eines anderen Flow-Objektes verbunden werden, wenn dieser Eingang den gleichen Datentyp hat. Um unterschiedliche Datentypen miteinander zu kombinieren, müssen diese in Zwischenschritten konvertiert werden. Folgende Datentypen werden unterstützt:

Datentyp	**Zulässiger Wertebereich**	**Farbe**	**Anmerkungen**
DIGITAL	0 und 1 (entsprechend LOW und HIGH)	Weiß	
NUMERISCH	-2100000000000..2100000000000	Türkis	Abhängig vom HMI-Element bzw. Flow-Objekt kann ein solcher numerischer Wert eine Fließpunktzahl oder eine ganze Zahl sein. Wird eine Fließpunktzahl auf ein Element geführt, welches nur Ganzzahlen verarbeitet, so wird diese Zahl gerundet.

ZEICHEN	Einfacher Text der alle druckbaren alphanumerischen Zeichen enthalten darf	violett	
BINÄR	Andere Daten mit variabler Größe, variabler Bedeutung und variablem internen Format	gelb	Binärdaten haben einen definierten Typ und eine definierte Länge. Diese Informationen sowie die Nutzdaten selber sind intern in so einem Binärpaket enthalten. Im Unterschied zu den anderen Datentypen muss der Benutzer bei Binärdaten darauf achten, dass der von einem Flow-Objekt abgegebene Binärdatentyp auch vom empfangenden Objekt verarbeitet werden kann, da nicht jedes Objekt mit jedem Binärtyp umgehen kann. So kann ein Binärdatenpaket beispielsweise Bilddaten enthalten. Dieser Typ von Binärdaten kann dann nur von Flow-Objekten verarbeitet werden, die Bilddaten erwarten. Werden diese Daten auf ein Objekt geleitet, welches nicht damit umgehen kann, so wird das Paket beim Empfänger verworfen und nicht weiter verarbeitet.

Die Farbe, die einen Datentyp repräsentiert, findet sich auch in den Pfeilen der Ein- und Ausgangssymbole im Flow-Editor wieder. Es dürfen also immer nur Ausgänge an Eingänge der gleichen Farbe angeschlossen werden. Eine solche Flow-Verbindung von einem Ausgang zu einem Eingang wird dabei durch Ziehen einer Linie beginnend vom Ausgang des Start-Objektes hin zum gewünschten Eingang des ziel-Objektes erstellt.

5.3.2.1 Flow-Objekte des Floweditors

5.3.2.1.1 Stand-Alone Flow-Objekte

5.3.2.1.1.1 Externe Laser-Flow-Objekte

G-Code Controller:

WARNUNG: Dieses Plug-In dient der Steuerung von Laser- oder anderem Maschinenequipment. Diese können gesundheitliche und andere Schäden verursachen. Vor Installation und Inbetriebnahme ist daher die Einhaltung aller relevanten Sicherheitsregeln zu gewährleisten. Es liegt im alleinigen Verantwortungsbereich des Anwenders, alle notwendigen und relevanten Sicherheitsregeln betreffend Installation und Betrieb des Systems zu jeder Zeit strikt einzuhalten.

Mit diesem Plug-In ist es möglich, Laser-Controller oder ähnliche Geräte anzusteuern, wenn diese G-Code (CNC) Daten über Ethernet akzeptieren. Hierbei handelt es sich um ein generisches Plug-In, welches nicht auf einen bestimmten Gerätetyp spezialisiert ist, sondern mit allen G-Code-kompatiblen Controllern funktionieren sollte. Innerhalb des Konfigurationsdialoges muss das Plug-In entsprechend den angeschlossenen und zu verwendendenden Geräten konfiguriert werden.

Allgemeine Konfigurationsparameter umfassen dabei:

- IP und Port – hier muss die IP und Portnummer des anzusteuernden Gerätes angegeben werden
- Achsenanzahl – legt fest, ob der Controller im 2D- oder 3D-Modus betrieben werden soll; diese Option hängt von den Fähigkeiten der angeschlossenen Geräte ab
- Feld linke Position / Feld obere Position – legt die linke obere Koordinate des Arbeitsbereiches fest, welcher vom Scanner verwendet werden soll

- Feldgröße – die Gesamtgröße des Arbeitsbereiches, welcher vom Scanner verwendet werden soll. Dieser Parameter spezifiziert zusammen mit der linken oberen Ecke den Bereich, innerhalb dessen alle Koordinaten der Eingangsvektordaten liegen müssen. Befinden sich Vorgabedaten außerhalb dieses Bereiches, so werden diese nicht bzw. am Rand des Arbeitsbereiches markiert.
- Initialisierungskommandos – hier ist es möglich, eigene, gerätespezifische G-Code-Kommandos einzutragen, welche bei der Initialisierung gesendet werden sollen
- Sprungmodus – hier kann ausgewählt werden, wie Sprünge bei abgeschaltetem Laser (bzw. abgeschaltetem Werkzeug) ausgeführt werden sollen. Dabei kann zwischen "G00" (schnellstmögliche Bewegung zum Zielpunkt auf nicht vorgegebenem Weg) oder "Sprunggeschwindigkeit" (normale Bewegung unter Verwendung der unten eingestellten Sprunggeschwindigkeit) gewählt werden
- X und Y tauschen – tauscht die X- und Y-Eingangskoordinaten, so dass die Werte für die X- und Y-Achse des Scanners vertauscht sind.
- Spiegelung X / Spiegelung Y – spiegelt die Eingangskoordinaten entlang der X- bzw. Y-Achse

Voreingestellt:

Hier können diverse Werte eingestellt werden, welche im Fall einer direkten Ansteuerung der Achsen X, Y und Z oder im Falle der Ansteuerung über Control-Daten zum Einsatz kommen:

- Sprunggeschwindigkeit – die Geschwindigkeit, mit der der Scanner seine Position ändert, wenn der Laser ausgeschaltet ist (für den Fall, dass der enstprechende Sprungmodus eingestellt wurde)
- Markiergeschwindigkeit – die Geschwindigkeit, mit der der Scanner seine Position ändert, wenn der Laser eingeschaltet ist; dieser Parameter wird nur im ControlRoom-Kontext beachtet und das nur dann, wenn dort keine binären Steuerungsdaten verwendet werden

Innerhalb eines ControlRoom-Projektes kann ein E1701 Scanner-Plug-In entweder einzeln über die separaten Eingänge (wie X, Y und Z) oder über den Control-Eingang angesteuert werden. Hier dürfen nicht beide Möglichkeiten gleichzeitig und ohne Synchronisation verwendet werden, da es zu undefinierten Ergebnissen führt, wenn sowohl Control-Streams verwendet werden als auch die anderen Eingänge mit sich möglicherweise widersprechenden Daten versorgt werden. Wird der Control-Eingang verwendet, so ist der BSY-Ausgang des Plug-Ins mit dem Plug-In zu verbinden, welches die Control-Daten erzeugt und ausgibt.

Folgende Ein- und Ausgänge werden vom Plug-In zur Verfügung gestellt:

Numeric IN0 (X) – die X-Position, zu welcher der Scanner bewegt werden soll. Wird an diesem Eingang ein neuer Wert gesetzt, so bewirkt das noch keine Bewegung, das passiert erst, wenn auch der Y- oder Z-Eingang gesetzt wird

Numeric IN1 (Y) – die Y-Position, zu welcher der Scanner bewegt werden soll. Ist das Plug-In zur Verwendung von zwei Achsen konfiguriert, so verursacht ein Setzten dieses Wertes eine Bewegung zu den Koordinaten x,y. Ist der Laser aus, so wird für diese Bewegung die Sprunggeschwindigkeit oder das Kommando G00 verwendet. Ist er an, wird die Markiergeschwindigkeit verwendet.

Numeric IN2 (Z) – die Z-Position, zu welcher der Scanner bewegt werden soll. Ist das Plug-In zur Verwendung von drei Achsen konfiguriert, so verursacht ein Setzten dieses Wertes eine Scannerbewegung zu den Koordinaten x,y,z. Ist der Laser aus, so wird für diese Bewegung die Sprunggeschwindigkeit oder das Kommando G00 verwendet. Ist der Laser an, wird die Markiergeschwindigkeit verwendet.

Digital IN3 (L) – schaltet den Laser an und aus

Numeric IN6 (CMD) – über diesen Eingang können Kommandos in Form von Zahlenwerten an das Plug-In gesendet werden, um damit zusätzliche Steueroperationen auszuführen. Derzeit werden folgende Kommandos unterstützt:

- 0 – stoppt den Markiervorgang unverzüglich und leert die Liste an möglicherweise noch gespeicherten Control-Kommandos innerhalb des Plug-Ins oder auf dem Scannercontroller

Binary IN7 (CTRL) – hier wird ein (zusammenhängender) Strom and Control-Daten erwartet, welcher Daten zur Bewegung des Scanners und zur Ansteuerung des Lasers enthält, ein solcher Datenstrom kann komplexe Markierinformationen enthalten, wie sie beispielsweise von BeamConstruct-Projekten stammen können

Digital OUT6 (BSY) – signalisiert, ob ein Markiervorgang aktiv ist (HIGH) oder ob der Scannercontroller alle Operationen beendet hat; dieser Ausgang muss mit dem BSY-Eingang desjenigen Plug-Ins verbunden werden, welches die Control-Daten erzeugt. Das ist notwendig, um verschiedene Bewegungsoperationen miteinander zu synchronisieren.

Dieses Plug-In befindet sich in flowplugins/libio_sc_gcode.

HALaser E1701D Scannercontroller:

WARNUNG: Dieses Plug-In dient der Steuerung eines Laser-Scanner-Systems. Laserstrahlen können gesundheitliche und andere Schäden verursachen. Vor Installation und Inbetriebnahme ist daher die Einhaltung aller relevanten Sicherheitsregeln zu gewährleisten. Es liegt im alleinigen Verantwortungsbereich des Anwenders, alle notwendigen und relevanten Sicherheitsregeln betreffend Installation und Betrieb des Systems zu jeder Zeit strikt einzuhalten. Darüber hinaus besteht die Möglichkeit, dass Laserequipment durch die Ansteuerung mit falschen Signalen beschädigt wird. Deshalb wird es dringend empfohlen, die von diesem Plug-In erzeugte Signale vor ihrer Verwendung zu überprüfen (z.B. mit einem Oszilloskop) um Fehlkonfigurationen oder sonstige Probleme vorab zu identifizieren. Das sollte vor jeder Erstinbetriebnahme und vor jeder Inbetriebnahme nach Softwareupdates gemacht werden.

Mit diesem Plug-In ist es möglich, einen E1701D Scannercontroller von HALaser Systems/OpenAPC Project Group zu verwenden und einen daran angeschlossenen Scankopf sowie den zugehörigen Laser in Echtzeit anzusteuern. Innerhalb des Konfiurationsdialoges müssen das Plug-In und die Scannerkarte entsprechend den angeschlossenen und zu verwendendenden Geräten konfiguriert werden.

Allgemeine Konfigurationsparameter umfassen dabei:

- IP oder serielle Schnittstelle - hier wird eingestellt, wie und wo auf den Controller zugegriffen werden kann. Im Fall einer Verbindung über Ethernet muss die IP der Karte angegeben werden (im Stil aaa.bbb.ccc.ddd). Im Fall einer Verbindung über USB ist der Name der seriellen Schnittstelle vorzugegeben ("COMx" für Windows, "/dev/ttyACMx" für Linux wobei "x" die Nummer der Schnittstelle ist, die zum Controller gehört)
- Ethernetpasswort - dieser Parameter muss nur im Falle einer Verbindung über Ethernet angegeben werden; hier kann ein Passwort vorgegeben werden, welches einen bestimmten Controller eindeutig identifiziert. Das gleiche Passwort muss dabei ebenfalls in der Konfiguration des entsprechenden E1701D Controllers angegeben werden. Wenn ein hier gesetztes Passwort nicht mit dem auf dem Controller konfigurierten übereinstimmt, so trennt dieser seine Verbindung automatisch und es ist nicht möglich, diesen zu benutzen.
- Lasertyp - der Typ des angeschlossenen Lasers; die hier vorgenommene Einstellung steht in direkter Beziehung zu den Laser-Tabpanes, in denen zusätzliche, lasertypspezifische Einstellungen vorgenommen werden können. Eine detaillierte Beschreibung von deren Eigenschaften und den zugehörigen Konfigurationsparametern liefert das Handbuch für die E1701 Scannercontroller sowie die Beschreibung des Laserherstellers.
- Power output - die Laserausgangsleistung kann zusätzlich über einen weiteren Ausgang kontolliert werden, welcher sich hier vorgeben lässt. ACHTUNG: die hier aufgeführten Ausgänge benötigen spezielle Varianten des E1701 Scannercontrollers und/oder spezielle Erweiterungskarten, welche optional sind. Detaillierte Auskunft über die verfügbaren Varianten und Erweiterungsboards sowie deren Ausgänge liefert das E1701 Handbuch.
- Korrekturfaktor Analogausgang: dieser Parameter wird nur bei Verwendung eines LP8 Erweiterungsboards genutzt. Dieses Board bietet einen Analogausgang A0, welcher im Bereich 0..5V arbeitet. Für diesen wird eine externe Stromversorgung mit exakt 5V benötigt, da die über den USB-Anschluss bereitgestellte Spannung oftmals weniger als 5V beträgt. Mit diesem Korrekturfaktor kann die Ausgangsspannung des Analogausganges korrigiert werden. Allerdings ist es natürlich nicht möglich, bei weniger als 5V Versorgung den Maximalwert auf 5V anzuheben.
- Achsenanzahl - legt fest, ob die Scannerkarte im 2D- oder 3D-Modus betrieben werden soll; diese Option hängt von den Fähigkeiten der Scannerkarte und des verwendeten Scankopfes ab
- Korrekturdatei - Pfad zu einer Korrekturtabelle, die für die Markieroperationen verwendet werden soll (und in der Regel vom Hersteller des Scankopfes zur Verfügung gestellt wird). Hier werden verschiedene Formate von der Karte direkt unterstüzt, so dass diese ohne vorherige Konvertierung verwendet werden können.

- Standby-Frequenz - die Frequenz, mit der der Laser im Standby-Modus angesteuert wird (abhängig vom gewählten Lasertyp)
- Standby-Länge - die Pulslänge, mit der der Laser im Standby-Modus angesteuert wird (abhängig vom gewählten Lasertyp)
- Feld linke Position / Feld obere Position - legt die linke obere Koordinate des Arbeitsbereiches fest, welcher vom Scanner verwendet werden soll
- Feldgröße - die Gesamtgröße des Arbeitsbereiches, welcher vom Scanner verwendet werden soll. Dieser Parameter spezifiziert zusammen mit der linken oberen Ecke den Bereich, innerhalb dessen alle Koordinaten der Eingangsvektordaten liegen müssen. Befinden sich Vorgabedaten außerhalb dieses Bereiches, so werden diese nicht bzw. am Rand des Arbeitsbereiches markiert.
- X uns Y tauschen - tauscht die X- und Y-Eingangskoordinaten, so dass die Werte für die X- und Y-Achse des Scanners vertauscht sind.
- Spiegelung X / Spiegelung Y - spiegelt die Eingangskoordinaten entlang der X- bzw. Y-Achse
- On-the-fly-Faktor X / On-the-fly-Faktor Y - legt den Faktor für on-the-fly-Operationen für die X- bzw. Y-Richtung fest; zur Verwendung dieser Funktionalität wird das Digi I/O Erweiterungsboard benötigt, Details dazu finden sich in der E1701-Beschreibung
- Gain X / Gain Y - linearer Korrekturfaktor für die X- bzw. Y-Richtung zum Strecken bzw. Stauchen der Ausgabe
- Rotation - dreht die Ausgabe um den hier vorgegebenen Winkel (in der Einheit Grad), als Rotationsmittelpunkt wird die Mitte des Arbeitsbereiches verwendet.
- Eingang "Laser bereit" - legt einen Digitaleingangfest, welcher für ein Signal "Laser bereit" benutzt werden kann. Wird diese Option verwendet, so kann eine Markieroperation nur dann gestartet werden, wenn an dem hier vorgegebenen Eingangspin ein HIGH-signal anliegt. Diese Funktion kann nur zusammen mit dem Digi I/O Erweiterungsboard verwendet werden.
- Firmware Debuglog - im Falle von Problemen mit der E1701-Karte oder im Fall, dass wärend des Betriebes der Karte die Fehler-LED aufleuchtet, kann dieser Parameter zur Fehlersuche herangezogen werden. Damit werden Loginformationen von der Controllerkarte geladen und zur weiteren Auswertung in eine Datei geschrieben. Um diese Funktionalität zu aktivieren, muss der Pfad zu einer Datei angegeben werden, in welche die Logdaten geschrieben werden können. Wird das Eingabefeld leer gelassen, so ist diese Funktion deaktiviert.
  ACHTUNG: Das Firmware-Logfile wird bei jedem Start neu erzeugt und alle vorhergehend geschriebenen Daten werden dabei gelöscht. Um vollständige Logdaten zu erhalten, sind deswegen die folgenden Schritte auszuführen:
  1. E1701 Scannercontroller einschalten (oder neu starten bzw. zurücksetzen, wenn dieser bereits läuft)
  2. die Applikation starten, die den E1701 Controller verwendet (zusammen mit einer Konfiguration, in der die Option für das Erzeugen des Debuglogs aktiviert ist)
  3. die Operationen ausführen, welche das unerwartete Verhalten verursachen oder zu einem Aufleuchten der Fehler-LED führen
  4. diese Operationen anhalten und die Applikation beenden
  5. das Firmware-Debuglog auswerten (oder zur weiteren Hilfestellung an HALaser Systems/OpenAPC Project Group senden)

Voreingestellt:

Hier können diverse Werte eingestellt werden, welche im Fall einer direkten Ansteuerung der Achsen X, Y und Z oder im Falle der Ansteuerung über Control-Daten (beide innerhalb eines controlRoom-Projektes) zum Einsatz kommen. Dieses Parameterpanel wird innerhalb BeamConstruct nicht angezeigt, da diese Werte dort nicht verwendet werden:

- Sprunggeschwindigkeit - die Geschwindigkeit, mit der der Scanner seine Position ändert, wenn der Laser ausgeschaltet ist
- Markiergeschwindigkeit - die Geschwindigkeit, mit der der Scanner seine Position ändert, wenn der Laser eingeschaltet ist

- Laser-Aus Verzögerung / Laser-Ein Verzögerung / Sprungverzögerung / Markierverzögerung / Polygonverzögerung - diese Pausen kommen während einer Markieroperation zum Einsatz und werden aktiv, wenn der Laser aus- oder eingeschaltet wird, wenn der Scanner zu einer neuen Position springt oder sich mit aktiviertem Laser dort hin bewegt oder wenn sich die Bewegungsrichtung während eines Markiervorganges ändert.

Alle folgenden Tabpanes enthalten laserspezifische Einstellmöglichkeiten, welche abhängig vom selektierten Lasertyp verwendet werden. Für weitergehende Informationen über diese Parameter und ihre Verwendung sind die Beschreibungen der jeweiligen Laser zu konsultieren.

Innerhalb eines ControlRoom-Projektes kann ein E1701 Scanner-Plug-In entweder einzeln über die separaten Eingänge (wie X, Y und Z) oder über den Control-Eingang angesteuert werden. Hier dürfen nicht beide Möglichkeiten gleichzeitig und ohne Synchronisation verwendet werden, da es zu undefinierten Ergebnissen führt, wenn sowohl Control-Streams verwendet werden als auch die anderen Eingänge mit sich möglicherweise widersprechenden Daten versorgt werden. Wird der Control-Eingang verwendet, so ist der BSY-Ausgang des Plug-Ins mit dem Plug-In zu verbinden, welches die Control-Daten erzeugt und ausgibt.

Folgende Ein- und Ausgänge werden vom Plug-In zur Verfügung gestellt:

Numeric IN0 (X) – die X-Position, zu welcher der Scanner bewegt werden soll. Wird an diesem Eingang ein neuer Wert gesetzt, so bewirkt das noch keine Bewegung, das passiert erst, wenn auch der Y- oder Z-Eingang gesetzt wird

Numeric IN1 (Y) – die Y-Position, zu welcher der Scanner bewegt werden soll. Ist das Plug-In zur Verwendung von zwei Achsen konfiguriert, so verursacht ein Setzten dieses Wertes eine Scannerbewegung zu den Koordinaten x,y. Ist der Laser aus, so wird für diese Bewegung die Sprunggeschwindigkeit verwendet, ist er an, wird die Markiergeschwindigkeit verwendet.

Numeric IN2 (Z) – die Z-Position, zu welcher der Scanner bewegt werden soll. Ist das Plug-In zur Verwendung von drei Achsen konfiguriert, so verursacht ein Setzten dieses Wertes eine Scannerbewegung zu den Koordinaten x,y,z. Ist der Laser aus, so wird für diese Bewegung die Sprunggeschwindigkeit verwendet, ist er an, wird die Markiergeschwindigkeit verwendet.

Digital IN3 (L) – schaltet den Laser an und aus

Numeric IN4 (PWR) – die Leistung, mit welcher der Laser betrieben werden soll, hier ist ein prozentualer Leistungswert im Bereich 0..100 anzugeben.

Numeric IN5 (FREQ) – die Frequenz, mit welcher der Laser während Markiervorgängen betrieben werden soll.

Numeric IN6 (CMD) – über diesen Eingang können Kommandos in Form von Zahlenwerten an das Plug-In gesendet werden, um damit zusätzliche Steueroperationen auszuführen. Derzeit werden folgende Kommandos unterstützt:

- 0 – stoppt den Markiervorgang unverzüglich und leert die Liste an möglicherweise noch gespeicherten Control-Kommandos innerhalb des Plug-Ins oder auf dem Scannercontroller

Binary IN7 (CTRL) – hier wird ein (zusammenhängender) Strom and Control-Daten erwartet, welcher Daten zur Bewegung des Scanners und zur Ansteuerung des Lasers enthält, ein solcher Datenstrom kann komplexe Markierinformationen enthalten, wie sie beispielsweise von BeamConstruct-Projekten stammen können

Digital OUT6 (BSY) – signalisiert, ob ein Markiervorgang aktiv ist (HIGH) oder ob der Scannercontroller alle Operationen beendet hat; dieser Ausgang muss mit dem BSY-Eingang desjenigen Plug-Ins verbunden werden, welches die Control-Daten erzeugt. Das ist notwendig, um verschiedene Bewegungsoperationen miteinander zu synchronisieren.

Binary OUT7 (IN) – binäre Control-Daten, welche den Zustand der Digitaleingänge der Scannerkarte widerspiegeln (benötigt Digi IO Erweiterungsboard). Diese Daten werden von einigen Plug-Ins benötigt, welche auf diese Eingänge reagieren und dementsprechend logisch mit diesen verbunden sein müssen (wie beispielsweise das Scannercontroller-Stepper-Plug-In).

Dieses Plug-In befindet sich in flowplugins/libio_e1701.

5.3.2.1.1.2 Externe Motion-Flow-Objekte

E1701M Schrittmotor-Controller

Mit diesem Plug-In können bis zu drei Achsen eines E1701M Schrittmotorcontrollers von HALaser Systems/OpenAPC Project Group angesteuert werden. Diese Achsen können unabhängig voneinander oder synchron über den Control-Eingang angesteuert werden. Im zweiten Fall ist es möglich, X/Y-Bewegungen in der Ebene oder dreidimensionale Bewegungen mit X/Y/Z durchzuführen. So kann dieses Plug-In beispielsweise einen XY-Tisch oder ein Maschinenwerkzeug ansteuern. Der Controller selbst kann per USB oder Ethernet angesprochen werden.

Innerhalb der Konfigurationspanels gibt es zwei Hauptbereiche. Das erste Panel bietet allgemeine Einstellungsmöglichkeiten, welche sich auf den Controller beziehen:

- IP oder serielle Schnittstelle - hier wird eingestellt, wie und wo auf den Controller zugegriffen werden kann. Im Fall einer Verbindung über Ethernet muss die IP der Karte angegeben werden (im Stil aaa.bbb.ccc.ddd). Im Fall einer Verbindung über USB ist der Name der seriellen Schnittstelle vorzugegeben ("COMx" für Windows, "/dev/ttyACMx" für Linux wobei "x" die Nummer der Schnittstelle ist, die zum Controller gehört)
- Ethernetpasswort - dieser Parameter muss nur im Falle einer Verbindung über Ethernet angegeben werden; hier kann ein Passwort vorgegeben werden, welches einen bestimmten Controller eindeutig identifiziert. Das gleiche Passwort muss dabei ebenfalls in der Konfiguration des entsprechenden E1701M Controllers angegeben werden. Wenn ein hier gesetztes Passwort nicht mit dem auf dem Controller konfigurierten übereinstimmt, so trennt dieser seine Verbindung automatisch und es ist nicht möglich, diesen zu benutzen.
- X und Y tauschen - diese Option ist nur für den Fall gültig, dass das Plug-In innerhalb eines ControlRoom-Projektes mit Koordinatenwerten (Control-Eingang) und nicht mit separaten Positionswerten X und Y versorgt wird. Ist diese Option gesetzt, so werden die X- und Y-Koordinate getauscht. Wird das Plug-In aus BeamConstruct heraus verwendet, so hat diese Option keine Auswirkung.
- Firmware Debuglog - im Falle von Problemen mit der E1701M-Karte oder im Fall, dass wärend des Betriebes der Karte die Fehler-LED aufleuchtet, kann dieser Parameter zur Fehlersuche herangezogen werden. Damit werden Loginformationen von der Controllerkarte geladen und zur weiteren Auswertung in eine Datei geschrieben. Um diese Funktionalität zu aktivieren, muss der Pfad zu einer Datei angegeben werden, in welche die Logdaten geschrieben werden können. Wird das Eingabefeld leer gelassen, so ist diese Funktion deaktiviert.
  ACHTUNG: Das Firmware-Logfile wird bei jedem Start neu erzeugt und alle vorhergehend geschriebenen Daten werden dabei gelöscht. Um vollständige Logdaten zu erhalten, sind deswegen die folgenden Schritte auszuführen:
  1. E1701M Motioncontroller einschalten (oder neu starten bzw. zurücksetzen, wenn dieser bereits läuft)
  2. die Applikation starten, die den E1701M Controller verwendet (zusammen mit einer Konfiguration, in der die Option für das Erzeugen des Debuglogs aktiviert ist)
  3. die Operationen ausführen, welche das unerwartete Verhalten verursachen oder zu einem Aufleuchten der Fehler-LED führen
  4. diese Operationen anhalten und die Applikation beenden
  5. das Firmware-Debuglog auswerten (oder zur weiteren Hilfestellung an HALaser Systems/OpenAPC Project Group senden)

Die Einstellmöglichkeiten innerhalb der Achsen-Panels gelten für jeweils eine spezifische Achse. Hier können für jede dieser Achsen unterschiedliche Einstellungen vorgenommen werden:

- Achse aktivieren - eine Achse wird von diesem Plug-In nur dann verwendet und kann demzufolge auch nur dann angesteuert und bewegt werden, wenn diese Checkbox gesetzt ist. Wird das Plug-In im Koordinatenmodus verwendet (Ansteuerung über den Control-Eingang bei dem anstatt von

Einzelpositionen für jede Achse XY- oder XYZ-Positionswerte gesendet werden), so müssen für den 2D-Betrieb (X und Y) die Achsen 1 und 2 aktiviert werden. Für den 3D-Betrieb (X, Y und Z) ist es entsprechend erforderlich, alle drei Achsen zu aktivieren.

- Achsenmodus - eine Achse kann abhängig von der gewünschen Anwendung in zwei verschiedenen Modi betrieben werden: planar oder radial. Im planaren Modus bewegt sie angeschlossene Maschinenteile über eine Entfernung (unter Verwendung der Maßeinheit mm), im radialen Modus wird eine Rotationsbewegung angesteuert (Maßeingeit Grad)
- Auflösung - dieser Wert stellt den Umrechungsfaktor zwischen den an den Motor gesendeten Inkrementalwerten und der tatsächlich ausgeführten Bewegung dar (Einheit inc/mm im planaren Modus, inc/Grad im radialen Modus). Der hier einzutragende Faktor hängt damit von der Inkrementalauflösung des Motors, der Schrittkonfiguration der Treiberstufe und der verwendeten Übersetzung/des eingesetzten Getriebes ab.
- Unteres Limit - das untere Limit (in der Einheit mm) ist ein Softwarelimit, welches die Minimalposition festlegt, bis zu der eine Achse bewegt werden darf. Im Fall des Radialmodus wird dieser Wert ignoriert.
  ACHTUNG: wenn eine Achse noch nicht referenziert wurde, ist dieses Limit nutzlos, da die tatsächliche Achsenposition noch nicht bekannt ist. D.h. es muss als erstes immer eine Referenzfahrt durchgeführt werden!
- Oberes Limit - das obere Limit (in der Einheit mm) ist ein Softwarelimit, welches die Maximalposition festlegt, bis zu der eine Achse bewegt werden darf. Im Fall des Radialmodus wird dieser Wert ignoriert.
  ACHTUNG: wenn eine Achse noch nicht referenziert wurde, ist dieses Limit nutzlos, da die tatsächliche Achsenposition noch nicht bekannt ist. D.h. es muss als erstes immer eine Referenzfahrt durchgeführt werden!
- Maximalgeschwindigkeit - dieser Parameter kann verwendet werden, um die durch Uservorgaben gesetzte Verfahrgeschwindigkeit zu begrenzen. Unabhängig davon, welche Geschwindigkeit im späteren Betrieb vorgegeben wird, wird dieser (hier in der Einheit mm/Sekunde für den planaren und Grad/Sekunde für den radialen Modus angegebene) Wert nie überschritten.
- Beschleunigungsmodus - legt die Form einer Beschleunigungsrampe fest. Hier kann zwischen linearem Modus (sanfte, gleichförmige Beschleunigung), exponentiellem Modus (starker aber am Ende der Beschleunigungsphase unsanfter Modus, bei dem der Motor über das Ziel hinaus schießen könnte) und S-förmigem Modus (sehr sanft aber nicht sehr stark beschleunigend) gewählt werden.
- Beschleunigungswert - Faktor zur Festlegung der Stärke der Beschleunigung in Abhängigkeit zum gewählten Beschleunigungsmodus
- Abbremsmodus - legt die Form einer Abbremsrampe fest. Hier kann zwischen linearem Modus (sanftes, gleichförmiges Abbremsen), exponentiellem Modus (starker aber am Anfang der Abbremsphase unsanfter Modus, bei dem der Motor Inkrementalwerte verlieren könnte) und S-förmigem Modus (sehr sanft aber nicht sehr stark abbremsend) gewählt werden.
- Abbremswert - Faktor zur Festlegung der Stärke des Abbremsvorganges in Abhängigkeit zum gewählten Abbremsmodus
- Stopp-Abbremswert - Faktor zur Festlegung der Stärke des Abbremsvorganges bei allen Stopp-Ereignissen (Stopp durch Endschalter, durch Referenzschalter oder durch Stopp-Kommando)
- Richtung invertieren - alle Achsen verwenden eine "positive" und eine "negative" Bewegungsrichtung, um die Bewegung zu beschreiben. Wenn diese Option gesetzt ist, werden diese Richtungen getauscht, was sinnvoll sein kann, um z.B. eine Richtungsangabe einfacher verständlich oder passend zur Maschinenbewegung zu machen.
- Referenziermodus - nach dem Einschalten oder wenn ein Bewegungsfehler aufgetreten ist, bei dem die Inkrementalposition verloren gegangen sein könnte, muss eine Referenzfahrt durchgeführt werden. Dafür ist an einer definierten Position ein Referenzschalter notwendig (siehe unten). Mit Hilfe dieses Parameters kann eine Sequenz von Bewegungen ausgewählt werden, welche bei der Referenzierung ausgeführt werden.
- Referenz-Timeout - hier kann eine Zeit (in der Einheit Sekunden) festgelegt werden, nach der eine Referenzfahrt abgebrochen wird, so fern diese bis dahin nicht erfolgreich beendet werden konnte.

- Eingang Referenzsignal - hier kann der Eingang festgelegt werden, an welchem der Referenzschalter angeschlossen ist
- Referenziergeschwindigkeit 1/2/3 - eine Referenzfahrt wird in bis zu drei Schritten ausgeführt (abhängig vom gewählten Referenziermodus). Die dabei verwendeten Verfahrgeschwindigkeiten sollten möglichst klein sein und/oder mit zunehmenden Referenzierschritten kleiner werden, um ein Überfahren des Referenzschalters möglichst klein zu halten. Mit diesen drei Parametern können die Referenzgeschwindigkeiten für jeden dieser Schritte vorgegeben werden. Geschwindigkeitswerte für Schritte, die laut Referenziermodus nicht verwendet werden, ignoriert die Software.
- Referenzposition - nach Beendigung der Referenzfahrt wird die aktuelle Position auf den hier vorgegebenen Wert (in mm oder Grad) festgelet
- Endschaltereingang 1/2 - neben den Softwarelimits können auch hardwaregestützte Begrenzungen verwendet werden, welche sich über Endschalter realisieren lassen. Werden solche Schalter verwendet, so wird mittels dieser beiden Parameter festgelegt, an welche Digitaleingänge der E1701M-Karte diese angeschlossen sind. Sobald an einem dieser Eingänge ein HIGH-Signal erkannt wird, wird die Bewegung dieser Achse unter Verwendung des oben beschriebenen Stopp-Abbremswertes angehalten.
- Endschalter automatisch verlassen - wurde ein Endschalter befahren, so sind auf der betroffenen Achse normalerweise keine weiteren Bewegungen möglich, bis der Schalter wieder verlassen wurde. Wird diese Option gesetzt, so bewegt sich eine Achse in so einem Fall bis zum vollständigen Verlassen des Endschalters automatisch wieder zurück, so dass anschließend weitere Verfahrbewegungen möglich sind. ACHTUNG: wenn diese Option eingeschaltet ist, kann es zu unerwarteten Verfahrbewegungen entgegen zur aktuellen Bewegungsrichtung kommen!
- Encoder - der E1701M Motioncontroller unterstützt bis zu zwei externe Encoder welche um 90 Grad phasenverschobene Pulse zur Positionsermittlung übertragen. Ist ein solcher Encoder mit dem Decodereingang 0 (Digitaleingänge 0 und 1) oder dem Decodereingang 1 (Digitaleingänge 2 und 3) verbunden, können diese hier für eine Achse aktiviert werden. Wird ein Decodereingang neu aktiviert, so wird die Achsenposition of 0 gesetzt. Anschließend ist es also erforderlich, eine neue Referenzfahrt durchzuführen.
  ACHTUNG: ein Encoder kann nur für genau eine Achse verwendet werden. Wird der selbe Eingang für mehr als eine Achse konfiguriert wird, so sind die Ergebnisse undefiniert.
- Encoderfaktor - dieser Faktor wird zur Ermittlung der tatsächlichen Inkrementalposition verwendet, hier ist das Verhältnis zwischen Encoderpulsen und Motorinkrementen anzugeben

Wird das Plug-In aus einem ControlRoom-Projekt heraus verwendet, so sind folgende Flow-I/Os verfügbar:

Numeric IN0 (POS) – absolute Position (in der Einheit mm oder Grad) zu welcher die Achse 1 bewegt werden soll; hier wird die Auflösung aus der oben beschriebenen Konfiguration verwendet, um die interne, inkrementelle Position des Motors zu berechnen. Wird hier ein Wert vorgegeben, so startet die Bewegung unverzüglich unter Verwendung des zuletzt am MODE-Eingang gesetzten Geschwindigkeitswertes

Numeric IN1 (MODE) – an diesem Eingang können mehrere Werte übergeben werden. Wird ein Wert größer als 0 gesetzt, so wird dieser als Geschwindigkeitswert für die nächste Bewegung von Achse 1 interpretiert. Ein Wert von 0 stoppt die Achsenbewegung unverzüglich und ein Wert von -1 startet eine Referenzfahrt.

Numeric IN2 (POS) – führt an der Achse 2 eine Bewegung wie bei IN0 beschrieben aus

Numeric IN3 (MODE) – legt eine Geschwindigkeit für Achse 2 fest oder steuert deren Bewegung ähnlich wie IN1

Numeric IN4 (POS) – führt an der Achse 2 eine Bewegung wie bei IN0 beschrieben aus

Numeric IN5 (MODE) – legt eine Geschwindigkeit für Achse 3 fest oder steuert deren Bewegung ähnlich wie IN1

Binary IN7 (CTRL) – Eingang für 2D (X and Y) oder 3D (X, Y and Z) Control-Daten zur Bewegungsteuerung; hierfür muss die zugehörige Anzahl an Achsen korrekt konfiguriert sein. Der Ausgang OUT7 gibt dabei jeweils passende, bewegungssynchrone Control-Daten aus, welche beispielsweise zur Ansteuerung eines Werkzeuges verwendet werden können.

Numeric OUT0 (POS) – dieser Ausgang stellt die aktuelle Position von Achse 1 zur Verfügung, während

einer Bewegung ändert sich dieser Wert fortlaufend

Numeric OUT1 (MODE) – dieser Statusausgang gibt 0 aus, wenn Achse 1 steht und ansonsten die aktuelle Verfahrgeschwindigkeit

Numeric OUT2 (POS) – Positionsausgang für Achse 1 (ähnlich wie to OUT0)

Numeric OUT3 (MODE) – Status-/Geschwindigkeitsausgang für Achse 1 (ähnlich wie OUT1)

Numeric OUT4 (POS) – Positionsausgang für Achse 2 (ähnlich wie to OUT0)

Numeric OUT5 (MODE) – Status-/Geschwindigkeitsausgang für Achse 2 (ähnlich wie OUT1)

Digital OUT6 (BSY) – dieser Ausgang stellt ein BUSY-Signal zur verfügung, welches mit einem HIGH-Wert signalisiert, ob das Plug-In derzeit noch Bewegungen ausführt oder nicht. Der Ausgang gibt erst dann wieder ein LOW-Signal aus, wenn sich keine der Achsen mehr bewegt und auch kein Verfahrkommando mehr in Abarbeitung ist. Dieser Ausgang muss mit dem BSY-Eingang desjenigen Plug-Ins verwendet werden, welches die binären Bewegungsdaten (für IN7) bereitstellt

Binary OUT7 (CTRL) – Ausgang für an IN7 bereitgestellte Control-Daten; diese werden hier synchron zur und abhängig von der aktuellen Bewegung ausgegeben, so dass sich damit beispielsweise ein Werkzeug abhängig von der aktuellen Position korrekt ansteuern lässt. Dieser Ausgang beachtet auch möglicherweise vorhandene Verzögerungen zum Ein- oder Ausschalten eines Werkzeuges. Sind diese Verzögerungen kleiner 0, so werden die Steuerungsdaten entsprechend lange vor Beginn der zugehörigen Bewegung an diesem Ausgang ausgegeben. Negative an/aus-Verzögerungen werden also korrekt behandelt.

Dieses Plug-In befindet sich in flowplugins/libio_e1701m_stepper.

5.3.3 Die Liste der Plugged Devices im OpenEditor

Im Flow-Editor ist es möglich, Plug-Ins einzufügen, welche direkt auf angeschlossene Hardware zugreifen. Dieser direkte Kommunikationsweg, bei dem keine klare Trennung zwischen den logischen Schichten einer Anwendung vorhanden ist, sollte für größere Projekte nicht verwendet werden.

Statt dieser direkten Kommunikation ist es empfehlenswert, solche Plug-Ins in einer eigenen logischen Ebene anzuordnen, welche sich unterhalb des OpenPlayers befinden. Die Kommunikation mit diesem findet dabei über den Interlock Server statt. Um ein solches Setup zu verwenden ist in einem ersten Schritt der Interlock Server in der Projektkonfiguration zu aktivieren. Dadurch wird in der Tab-Pane „Plugged Devices“ eine Liste mit Plug-Ins aktiv, in der all diejenigen Plug-Ins aufgelistet und konfiguriert werden können, welche in der Lage sind, mit Hardware zu kommunizieren.

Flow-Objekte können dabei auf die gleiche Weise wie im Flow-Editor hinzugefügt werden: entweder über das Kontextmenü (Klick mit der rechten Maustaste) oder über die Foldbar auf der rechten Seite des Fensters.

Wurde ein Flow-Objekt der Liste hinzugefügt, so wird dessen Symbol, ein eindeutiger Name, eine eindeutige Nummer und die Kategorie, zu der das Plug-In gehört, angezeigt. Die Nummer kann dabei nicht verändert werden, sie wird durch das Programm vorgegeben. Sie ist wichtig, wenn das Projekt später dem OpenPlugger übergeben wird, sie identifiziert das durch den OpenPlugger zu verwaltende Plug-In.

Im Unterschied zum Flow-Editor ist es in dieser Liste nicht möglich, Flow-Verbindungen zu erstellen. Flow-Objekte, die hier hinzugefügt wurden, laufen in je einer eigenen Instanz des OpenPluggers, welcher alle Ein- und Ausgänge des Plug-Ins im Interlock Server zur Verfügung stellt. D.h. Eingangsdaten zum Plug-In werden vom Interlock Server an dieses gesendet während der OpenPlugger Ausgangsdaten zurück an den Server schickt.

5.4 Benutzung des OpenDebuggers

Der OpenDebugger ist eine spezielle Umgebung für ein ControlRoom-Projekt. Ähnlich dem OpenPlayer kann hier ein Projekt geladen und ausgeführt werden. Im Unterschied dazu werden im OpenDebugger aber verschiedene Möglichkeiten angeboten, den Datenfluss und Programmablauf eines Projektes zu analysieren

und mögliche Probleme zu finden:

- Es können Abbruch-Bedingungen definiert werden, bei deren Eintreten der gesamte Programmablauf gestoppt wird
- Der Programmablauf kann in Einzelschritten abgearbeitet werden, so dass jede Änderung von Daten und Zuständen exakt nachvollzogen werden kann
- Fehler und Warnungen werden zusammen mit dem Namen des verursachenden Flow-Objektes angezeigt
- Die Ausgänge von Flow-Objekten und die dort anstehenden Daten können überwacht und angezeigt werden
- Die Kommunikation mit dem Interlock Server kann angezeigt werden

Der OpenDebugger muss mit dem Pfad zum auszuführenden Projekt als Parameter gestartet werden. Wird er aus dem OpenEditor heraus gestartet, wird automatisch das dort aktuell geladene Projekt verwendet.

Der OpenDebugger öffnet beim Start zwei Fenster: das HMI-Fenster mit dem Benutzerinterface wie es auch auf dem Zielsystem angezeigt werden würde und ein zusätzliches Debuggerfenster, über welches auf die Funktionen des OpenDebugger zugegriffen werden kann.

5.5 Benutzung des OpenPlayer

Der OpenPlayer stellt eine Umgebung zum Ausführen eines ControlRoom-Projektes auf dem gewünschten Zielsystem zur Verfügung. Dieser ist dabei so schlank und effizient wie möglich und bietet deswegen keinerlei spezielle Funktionen oder Debuggingmöglichkeiten. Er wird unter Angabe des kompletten Pfades zur Projektdatei als Parameter gestartet. Der OpenPlayer lädt dann die Projektdatei und alle benötigten Plug-Ins, initialisiert diese und startet den Programmablauf. Verwendet das Projekt den Interlock Server und externe Plug-Ins lokal (d.h. auf dem gleichen Rechner) so werden sowohl der Interlock Server als auch die benötigten OpenPlugger-Instanzen mit den zugehörigen Plug-Ins ebenfalls gestartet.

Der Player bietet keine eigene Möglichkeit, einen Programmablauf zu stoppen. Ist im ControlRoom-Projekt selber keine entsprechende Funktion vorgesehen, so muss der Player mit den Mitteln des verwendeten Betriebssystems beendet werden (TaskManager, kill-Kommando, ...).

Für Windows-Betriebssysteme wird empfohlen, den OpenPlayer exklusiv an Stelle des Windows-Explorers zu starten, um dem Benutzer keinerlei Zugriffsmöglichkeiten auf das darunterliegende Betriebssystem zu geben (Details darüber sind in der MSDN zu finden). Dadurch werden Startmenü, Taskbar, Desktop-Popupmenüs und anderes vollständig blockiert und ein unberechtigter Benutzer kann keine Änderungen und Einstellungen am System vornehmen. Zugriff auf das System kann dann innerhalb des ControlRoom-Projektes über das “Execute Command” Plug-In ermöglicht werden. Das sollte natürlich nur Benutzern mit höheren Rechten als dem Standard-Operator erlaubt werden.

Für Linux-Systeme ist die Situation ähnlich, hier sollte ein Windowmanager ausgewählt werden, welcher keinen Zugriff auf das System erlaubt und der den OpenPlayer als einzige Applikation schon beim Start ausführt. Zugriff auf das System ist auch hier wieder auf dem gleichen Weg über das “Execute Command” Plug-In möglich, welches dann beispielsweise eine Konsole öffnen könnte.

5.5.1 OpenPlayer Kommandozeilenparameter

Der OpenPlayer kann über die Kommandozeile oder aus einem Skript heraus mit folgender Syntax und Parametern ausgeführt werden:

```
OpenPlayer.exe [-b] [-x xpos] [-y ypos] <projectname>
```

Oder bei nicht-Windows-Systemen:

```
OpenPlayer [-b] [-x xpos] [-y ypos] <projectname>
```

Die Parameter im einzelnen:

- `OpenPlayer` oder `OpenPlayer.exe` ist der Name der Applikation selbst und muss immer angegeben werden
- `-b` ist ein optionaler Parameter, welcher das HMI-Fenster mit einem Rahmen ausstattet, so dass der Benutzer dieses anklicken und an eine andere Position ziehen kann
- `-x` ist ein optionaler Parameter zur Festlegung der X-Position des geöffneten Playerfensters welche normalerweise 0 ist (linke Seite des Bildschirmes); nach diesem Parameter ist die gewünschte X-Position in der Einheit Pixel anzugeben
- `-y` ist ein optionaler Parameter zur Festlegung der Y-Position des geöffneten Playerfensters welche normalerweise 0 ist (obere Seite des Bildschirmes); nach diesem Parameter ist die gewünschte Y-Position in der Einheit Pixel anzugeben
- `projectname` ist ein zwingend benötigter Parameter und gibt den Pfad zum auszuführenden ControlRoom-Projekt an (.apcx-Datei) welches ausgeführt werden soll

5.6 Benutzung des OpenHPlayer

Der OpenHPlayer ist eine spezielle Umgebung zum Ausführen von ControlRoom-Projekten. Dieser erwartet als Kommandozeilenparameter den Pfad zum auszuführenden Projekt. Im Unterschied zum OpenPlayer hat der OpenHPlayer (H für “headless”, also eigentlich “ohne Monitorausgabe”) keinerlei grafisches Benutzerinterface und kann deswegen auch nur Flow-Objekte verwalten und ansteuern, nicht aber HMI-Elemente. Enthält ein Projekt HMI-Elemente, so werden diese also vollständig ignoriert. Abgesehen von diesem Unterschied verhält sich der OpenHPlayer exakt wie der OpenPlayer.

5.7 Benutzung des Interlock Servers

Beim OpenIServer handelt es sich um einen speziellen Server, der vollständig im Hintergrund läuft und kein Benutzerinterface besitzt. Das „I“ steht dabei für „Interlock“, allerdings kann dieser Server für deutlich mehr Anwendungen als nur Software-Interlocks benutzt werden. Er bietet eine Schnittstelle, um ein laufendes Projekt von außen zu kontrollieren und zu beeinflussen. Es ist also möglich, eigene Software anzubinden, welche verschiedene Aufgaben wie z.B. die Kontrolle über die Ablaufsteuerung übernimmt.

Auf die Daten des OpenIServer kann auf folgenden Wegen zugegriffen werden:

- Aus der Hauptapplikation (OpenDebugger oder OpenPlayer/OpenHPlayer), wenn Objekte so konfiguriert sind, dass die Zustände ihrer Ausgänge an den Interlock Server gesendet werden und/oder dass ihre Eingänge mit Daten vom Server beschrieben werden
- Aus einer eigenen Applikation heraus, welche die liboapc und deren Interlock-Server-Funktionen verwendet (zu Details siehe das OpenSDK und das zugehörige Entwicklerhandbuch)
- Über ein LUA-Skript oder eine Anweisungsliste (IL, Instruction List)

Die vom OpenIServer ausgeführten Operationen, welche all diese Funktionalität ermöglichen, lassen sich dabei einfach zusammenfassen:

1. Alle Werte werden unter Verwendung eines eindeutigen Bezeichners gespeichert (ein Name, der gleich dem Namen des korrespondierenden HMI-Elementes ist oder ein anderer, eindeutiger Name)

2. Wann immer neue Daten mit einem neuen Bezeichner zum Server geschickt werden, so werden auch diese gespeichert und alle verbundenen Clients werden über diese neuen Daten informiert
3. Wann immer neue Daten für einen bereits bekannten Knoten mit bekanntem Bezeichner an den Server gesendet werden, so werden die bisherigen Daten überschrieben und alle verbundenen Clients erhalten diese neuen Daten

Zusätzlich gelten folgende Regeln:

- Jeder Datensatz (der aus einem Bezeichner und den zugehörigen Daten besteht), ist genauso wie ein Flow-Objekt in der Hauptapplikation organisiert: er kann bis zu 8 einzelne Daten von jeweils einem der Datentypen DIGI, NUM, CHAR oder BIN (digital, numerisch, Zeichen oder binär) enthalten
- Eine Änderung eines solchen Datentyps während der Laufzeit ist nicht möglich, wurde einmal ein Wert mit einem bestimmten Typ gesetzt, so werden nur noch Daten des gleichen Typs akzeptiert
- Ein bestehender Datensatz kann während der Laufzeit nicht gelöscht werden

5.7.1 Interlock Server Modi

Der Interlock Server kann in verschiedenen Modi betrieben werden. Der gewünschte Modus kann in den Projekteinstellungen gewählt werden und hängt davon ab, welche Funktionalitäten des Interlock Server genutzt werden sollen. Die möglichen Operationsmodi werden im folgenden beschrieben.

Achtung: Ein laufender Interlock Server wird nicht dadurch beendet, dass sich alle Clientverbindungen von ihm trennen. In diesem Fall löscht er zwar alle intern gehaltenen Daten, bleibt selbst aber aktiv. Wird jetzt ein Projekt mit einem anderen Operationsmodus gestartet, so wird dieser vom noch laufenden Interlock Server NICHT automatisch übernommen, dieser muss zuvor erst beendet werden.

5.7.1.1 Einfacher Lokaler Modus

Dieser Betriebsmodus ist der Einfachste, Hier starten OpenPlayer bzw. OpenDebugger genau eine Instanz des OpenIServer automatisch. Dieser Interlock Server läuft auf dem gleichen Host wie das Projekt, so dass sich externe Applikationen ebenfalls auf diesen Host verbinden müssen. In diesem Modus sind sonst keine weiteren Aktionen notwendig.

5.7.1.2 Gespiegelter Lokaler Modus

In diesem Modus werden vom OpenPlayer oder OpenDebugger zwei Instanzen des OpenIServers gestartet. Eine dieser Instanzen wird sofort aktiv und kann in bekannter Weise verwendet werden. Die andere Instanz ist nicht von außen zu erreichen und verbindet sich ebenfalls mit dem aktiven Interlock Server um alle dort vorgehaltenen Daten zu spiegeln.

Sobald die erste Instanz des Interlock Servers unerwartet beendet wird, aktiviert der Zweite seinen Serversocket um mit den gespiegelten Daten von außen erreichbar zu sein und startet wiederum eine weitere Instanz des OpenIServer, damit diese ab sofort als Spiegelserver arbeitet.

Im Moment des automatischen Serverwechsels verlieren alle externen Applikationen ihre Verbindung zum bisherigen, aktiven Interlock Server. Diese müssen sich jetzt lediglich erneut auf die gleiche IP verbinden um mit dem jetzt eingesprungenen Server weiter zu arbeiten. Standardapplikationen wie Debugger, Player oder OpenPlugger führen diesen Reconnect automatisch aus.

5.7.1.3 Einfacher Modus über Netzwerk

Dieser Modus entspricht dem einfachen lokalen Modus mit einem Unterschied: der Interlock Server läuft auf einem anderen Host im Netzwerk. Deshalb ist es dem OpenPlayer/OpenDebugger nicht möglich diesen sowie die möglicherweise benötigten OpenPlugger-Instanzen automatisch zu starten. Aus diesem Grund ist eine eigene Startprozedur erforderlich, welche die Teilkomponenten des Systems in dieser Reihenfolge startet:

1. Der Interlock Server muss auf dem Host gestartet werden, dessen IP in den Projekteinstellungen angegeben wurde
2. Alle OpenPlugger-Instanzen müssen unter Verwendung des Projektfiles gestartet werden, welches auch dem OpenPlayer übergeben wird und welches die Informationen über den Host des Interlock Server enthält
3. Alle externen Applikationen, welche auf den OpenIServer zugreifen und Software-Interlocks realisieren, müssen gestartet werden
4. Der OpenPlayer muss mit dem gleichen Projektfile wie in Punkt 2 verwendet gestartet werden
5. Alle externen Applikationen, welche auf den OpenIServer zugreifen und automatische Sequenzen ausführen, müssen gestartet werden

5.7.1.4 Redundanter Modus über Netzwerk

Hierbei handelt es sich um den sichersten, aber auch um den komplexesten Betriebsmodus. Er erweitert den gespiegelten lokalen Modus um die Möglichkeit, die beiden Interlock Server auf unterschiedlichen Systemen im Netzwerk zu betreiben. Damit wird es möglich, verteilte Anlagen auch bei einem Hardwareausfall weiter zu betreiben. Auch bei diesem Modus ist eine festgelegte, manuelle Startprozedur notwendig, da die einzelnen Komponenten nicht mehr automatisch ausgeführt werden können:

1. Der erste Interlock Server muss auf dem Host gestartet werden, der in den Projektparametern auch als erstes angegeben wurde. Das geschieht mit dem Kommandozeilenparameter “-r <ip>” wobei “ip” die IP des anderen, jetzt noch nicht gestarteten Servers ist. Dieser Interlock Server wird automatisch der aktive Server.
2. Der zweite Interlock Server ist auf dem in den Projekteinstellungen angegebenen Host zu starten. Das geschieht mit dem Kommandozeilenparameter “-r <ip>” wobei “ip” die IP des anderen, bereits gestarteten Servers ist. Dieser Interlock Server wird automatisch der Spiegelserver.
3. Alle OpenPlugger-Instanzen werden mit dem Projektfile gestartet, in dem die IPs der beiden Interlock Server angegeben sind. Diese versuchen, sich mit beiden IPs zu verbinden und verwenden dann automatisch den aktiven Server.
4. Externe Applikationen, die Software-Interlocks implementieren, werden gestartet. Diese Applikationen müssen ebenfalls versuchen, beide IPs zu erreichen. Da nur der aktive Interlock Server erreichbar ist, ist eine Verbindung zu diesem herzustellen.
5. Der OpenPlayer ist mit dem bereits in Punkt 3 verwendeten Projekt zu starten, dieser verbindet sich ebenfalls automatisch mit dem aktiven Interlock Server, indem Verbindungen zu beiden IPs durchprobiert werden.
6. Externe Applikationen, die automatische Sequenzen implementieren, werden gestartet. Diese Applikationen müssen ebenfalls versuchen, beide IPs zu erreichen. Da nur der aktive Interlock Server erreichbar ist, ist eine Verbindung zu diesem herzustellen.

Wenn das gesamte System läuft und es zu einem Problem mit dem aktiven Interlock Server kommt, aktiviert der Spiegelserver seinen Serversocket und übernimmt die Aufgaben des aktiven Servers.

Externe Applikationen, welche sich mit diesem Server verbunden hatten, verlieren diese Verbindung und müssen sich jetzt auf den Server auf der anderen IP verbinden, um mit den dort gespiegelten Daten weiter zu arbeiten. Standardapplikationen wie der OpenPlayer oder die OpenPlugger machen das automatisch.

Sobald das Problem mit dem ersten Interlock Server behoben ist, ist dieser wieder mit der Kommandozeilenoption “-r <ip>”, zu starten. Er verbindet sich mit dem jetzt aktiven Interlock Server und wird selbst zum Spiegelserver.

5.8 Benutzung des OpenPlugger

Der OpenPlugger ist eine Applikation, welche in Systemarchitekturen mit Interlock Server und ausgelagerten Plug-Ins für Hardwarezugriffe Verwendung findet. Dieser erfüllt folgende Aufgaben:

- Laden von genau einem speziellen Plug-In aus der verwendeten ControlRoom-Projektdatei
- Kommunikation mit dem Plug-In (Initialisierung, Daten senden und empfangen,...)
- Kommunikation mit dem Interlock Server (vom Interlock Server empfangene Daten zum Plug-In senden und anders herum)

Der OpenPlugger kann also nur in Systemen mit Interlock Server verwendet werden. Wird ein lokaler Interlock Server verwendet, wird der OpenPlugger für jedes betroffene Plug-In vom OpenPlayer automatisch gestartet. Handelt es sich aber um ein verteiltes System, bei dem der Interlock Server auf einem anderen Host läuft, so sind die OpenPlugger unter Verwendung folgender Parameter manuell zu starten:

- **-p <projektdatei>** - legt den Pfad zur Projektdatei fest; hier muss exakt das gleiche Projekt wie bei allen anderen OpenPlugger-Instanzen sowie das auch vom OpenPlayer/OpenDebugger verwendete Projekt angegeben werden; zusätzlich ist es notwendig, dass die IP(s) der/des Interlock-Servers, die im Projekt angegeben sind, von diesem Host aus erreichbar ist/sind
- **-n <plug_in_name>** - der eindeutige Name des Plug-Ins, das von diesem OpenPlugger verwaltet werden soll und wie er in der Plug-In-Liste im Editor angegeben wurde
- **-i <id>** - die eindeutige Nummer des von diesem OpenPlugger zu verwaltenden Plug-Ins; wenn dieser Parameter verwendet wird, so muss der Plug-In-Name (option –n) nicht angegeben werden
- **-V** – veranlasst den OpenPlugger, zusätzliche Informationen in der Konsole auszugeben, um mögliche Probleme identifizieren zu können
- **-h** – zeigt eine kurze Liste aller vom OpenPlugger unterstützten Kommandozeilenparameter und ihre Bedeutung an

Zum Starten des OpenPluggers sind die Optionen **-p** (Projektdatei) und **-n** oder **-i** zwingend anzugeben, um das Plug-In aus dem Projekt zu identifizieren, welches von diesem OpenPlugger verwaltet werden soll.

6 CNConstruct

Bei CNConstruct handelt es sich um eine generische CAD-Applikation zur Erstellung von Bearbeitungssabläufen. Da diese im Wesentlichen die gleichen Basisfunktionalitäten wie das um laserspezifische Funktionen erweiterte BeamConstruct handelt, sei für eine Beschreibung auf das folgende Kapitel verwiesen.

6.1 CNConstruct modifizieren

CNConstruct kann auf regulärem Wege so verändert werden, dass es zusammen mit Maschinen oder zusätzlichem Equipment unter einem komplett eigenen Namen und Aussehen ausgeliefert werden kann. Folgende Eigenschaften können modifiziert werden:

- Der Name der Software
- Die Homepage im “About”-Dialog
- Das Logo, das beim Programmstart und im “About”-Dialog angezeigt wird
- Das in der Taskbar angezeigt Icon der Applikation (wird nicht auf allen Betriebssystemen unterstützt)
- Die Copyright-Information im “About”-Dialog; diese werden zu den bestehenden Copyright-Daten hinzugefügt
- Die Bilder der Toolbar und des Elementtrees

Um die Software auf diese Art zu modifizieren, wird eine einfache Textdatei im UTF-8-Format mit dem Namen `cnc.oem` benötigt, welche im Installationsverzeichnis der Software abgelegt werden muss (für Windows typischerweise C:\Program Files\OpenAPC, auf Linux-Systemen in /usr/local/openapc). In dieser Datei enthält jede Zeile einen festen Wert:

1. Der Name der Software
2. Der URL der im “About”-Dialog anzuzeigenden Homepage
3. Pfad zu einer Bilddatei im PNG-Format, welches als Logo angezeigt werden soll
4. Pfad zu einer kleinen Bilddatei im PNG-Format, welches als Icon angezeigt werden soll
5. Copyrighttext, der im “About”-Dialog angezeigt werden soll

Die Bilder in der Toolbar sowie im Elementtree können ebenfalls ausgetauscht werden, hier kommen zwei unterschiedliche Methoden zum Einsatz.

Um die Toolbar-Symbole zu ersetzen werden für jedes Plug-In zwei Bilder im PNG-Format benötigt, welche in den Installationsverzeichnissen dieser Plug-Ins abgelegt werden müssen. Der Basisname der Bilder ist dabei jeweils identisch mit dem Plug-In-Namen (ohne Dateinamenserweiterung .DLL oder .so). Für das Toolbar-Symbol muss dem Dateinamen ein “1.png” hinten angestellt werden, für das kleinere, zugehörige Symbol im Elementtree entsprechend ein “2.png”.
Als Beispiel: Die Symbole des Barcode-Plug-Ins sollen ersetzt werden. Dieses kann im Ordner “priplugins” unter dem Namen “libio_pri_barcode.dll” bzw. “libio_pri_barcode.so” gefunden werden. Es werden an dieser Stelle also ein Bild “libio_pri_barcode1.png” mit einer Größe von 25x25 Pixeln und ein Bild “libio_pri_barcode2.png” mit einer Größe von 15x15 Pixeln benötigt.

Um alle anderen Bilder, welche nicht direkt zu einem Plug-In gehören, zu ersetzen, müssen PNG-Dateien unter folgenden Namen im Unterordner “icons” abgelegt werden:

- “newprj.png” für das Symbol “Neu“
- “openprj.png” für das Symbol “Projekt laden”
- “saveprj.png” für das Symbol “Projekt speichern”

- “saveprjas.png” für das Symbol “Projekt speichern als”
- “quit.png” für das Symbol “Applikation beenden”
- “zoomin.png” für das Symbol “Hereinzoomen”
- “zoomout.png” für das Symbol “Herauszoomen”
- “zoomworkarea.png” für das Symbol “Auf Arbeitsbereich zoomen”
- “zoomselected.png” für das Symbol “Auf selektiertes Element zoomen”
- “pointdrag.png” für das Symbol “Punktmodus”

7 BeamConstruct

7.1 Sicherheitshinweise

Diese Software dient der Steuerung eines Laser-Scanner-Systems. Laserstrahlen können gesundheitliche und andere Schäden verursachen. Vor Installation und Inbetriebnahme ist daher die Einhaltung aller relevanten Sicherheitsregeln zu gewährleisten. Es liegt im alleinigen Verantwortungsbereich des Anwenders, alle notwendigen und relevanten Sicherheitsregeln betreffend Installation und Betrieb des Systems zu jeder Zeit strikt einzuhalten.

Darüber hinaus besteht die Möglichkeit, dass Laserequipment durch die Ansteuerung mit falschen Signalen beschädigt wird. Deshalb wird es dringend empfohlen, die von dieser Software erzeugten Signale vor ihrer Verwendung zu überprüfen (z.B. mit einem Oszilloskop) um Fehlkonfigurationen oder sonstige Probleme vorab zu identifizieren. Das sollte vor jeder Erstinbetriebnahme und vor jeder Inbetriebnahme nach Softwareupdates gemacht werden.
„Softwareupdate“ umfasst nicht nur Veränderungen im Softwarestand des OpenAPC-Paketes sondern auch davon abhängende Software wie z.B. Firmware oder Treiber von Scannerkarten.

7.2 Überblick

BeamConstruct ist eine auf Lasermarkierprozesse spezialisierte CAD-Anwendung. Die damit erstellten Projekte können zusammen mit einem Laserscannersystem direkt aus BeamConstruct heraus zur Materialbearbeitung per Laser verwendet werden. Zusätzlich ist es möglich, diese Projektdaten in verschiedenen Formaten zu speichern um sie dann aus einer ControlRoom-Basierten HMI-Anwendung/Prozessteuerung heraus ohne Einsatz von BeamConstruct selbst zu nutzen und um sie z.B. in komplexere Maschinensteuerungen zu integrieren. D.h. BeamConstruct – welches eigentlich nur einen komplexen Editor darstellt – muss auf dem Zielsystem mit der Maschinensteuerung nicht unbedingt vorhanden sein.

7.3 Position innerhalb des Gesamtsystems

BeamConstruct ist kein elementar wichtiger Bestandteil des OpenAPC-Paketes. Darüber hinaus benötigt das OpenAPC-Paket BeamConstruct auch dann nicht, wenn ein ControlRoom-Projekt Daten aus BeamConstruct verarbeitet und Markieroperationen ausführt. BeamConstruct ist also mehr ein Tool zum Erstellen von Produktionsdaten und weniger ein Tool zur Produktion selbst – auch wenn es möglich ist, Produktionsabläufe aus BeamConstruct heraus zu steuern.

D.h. ein regulärer Ablauf in der Verwendung der Software könnte so aussehen:

1. Die benötigten Prozessdaten werden in BeamConstruct erstellt, das kann beispielsweise abseits der Produktionsräume in z.B. einem Konstruktionsbüro passieren.
2. Diese Daten werden gespeichert oder exportiert.
3. Die so gespeicherten oder exportierten Daten werden zur Produktion transferiert, in der ein passendes ControlRoom-Projekt den Ablauf steuert.
4. Dieses ControlRoom-Projekt lädt die von BeamConstruct kommenden Daten und verwendet diese intern, um das angeschlossene Lasersystem anzusteuern und die Materialien in der gewünschten Art und Weise zu bearbeiten.

Optional kann zwischen den Schritten 2 und 3 noch an den Projektdaten feinjustiert werden. Das ist mit einer anderen (oder der gleichen) BeamConstruct-Installation möglich, wenn diese an eine ähnliche Laseranlage angeschlossen ist. In diesem Fall kann die Möglichkeit genutzt werden, direkt aus BeamConstruct heraus zu markieren und die Ergebnisse der Markieroperationen gleich zu begutachten und entsprechende

Änderungen am Projekt vorzunehmen.

7.4 Schnellstartanleitung für BeamConstruct

Dieser Abschnitt beschreibt, wie BeamConstruct im direkten Modus verwendet werden kann – d.h. direkt aus BeamConstruct heraus und ohne Verwendung eines separaten ControlRoom-HMI-Projektes. Er bietet einen kurzen Überblick über die grundlegende Konfiguration der Software und wie man schnell zu ersten, einfachen Ergebnissen kommen kann:

1. **SICHERHEITSCHECK**: Die folgenden Schritte beschreiben die Konfiguration von BeamConstruct und wie es möglich ist, Laserequipment direkt aus dieser Software heraus zu anzusteuern. Aus diesem Grund ist vor und während sämtlicher Schritte auf die Lasersicherheit und auf das Vorhandensein funktionierender technischer Sicherheitseinrichtungen zu achten.
2. **Konfiguration**: Als erstes muss die Scannerkarte gewählt und eingestellt werden, welche im Folgenden verwendet werden soll. Das geschieht im Menü “Project”, Menüpunkt “Project settings…” und im sich dann öffnenden Dialog im Panel “Scanner”. In der Auswahlliste „Scanner Card“ ist der gewünschte Scannercontroller auszuwählen und anschließend durch Betätigung des Buttons „Configure“ zu konfigurieren. Hierbei sind alle Scannercontroller-spezifischen Einstellungen entsprechend der Beschreibung des Herstellers vorzunehmen (hier müssen in der Regel Dinge wie Firmware, Korrekturdatei, Größe des Arbeitsbereiches, Anzahl der Achsen und ähnliches vorgegeben werden).
3. **Speichern der Konfiguration**: Sind alle Einstellungen korrekt vorgenommen worden, so werden diese Konfigurationsdialoge jeweils durch Drücken der Buttons “OK” verlassen. Um diese Parameter auch nach dem nächsten Programmstart wieder zur Verfügung zu haben, empfiehlt es sich, im Menü „Project“ den Menüpunkt “Save as default configuration” auszuwählen.
4. **Geometrien erzeugen**: Am oberen Bildschirmrand befinden sich in der Toolbar verschiedene Symbole mit geometrischen Figuren. Hier ist für dieses Beispiel das blaue Dreiecks-Symbol anzuklicken und das Dreieck mit drei Klicks an unterschiedlichen Positionen im Zeichenbereich unterhalb der Toolbar zu positionieren. Jeder dieser drei Mausklicks legt dabei einen der Dreiecks-Eckpunkte fest, zwischen diesen muss die Maustaste nicht festgehalten werden.
5. **Bestehende Geometrien interaktiv modifizieren**: Wurde das Dreieck erfolgreich erstellt, wird es in selektiertem Zustand dargestellt, was durch einen blauen, rechteckigen Rahmen gekennzeichnet wird. Dieser Rahmen kann verwendet werden, um das Dreieck zu verändern. Wird es mit der Maus innerhalb der im Rahmen enthaltenen farbigen Quadrate erfasst und gezogen, so lassen sich auf diesem Wege die Größe (graue Quadrate) und seine Rotation (rotes Quadrat) verändern. Mittels der türkisen Quadrate lässt es sich in horizontaler oder vertikaler Richtung neigen (das Ergebnis dieser Operation ist bei rechteckigen Geometrien besser zu erkennen). Wird das Dreieck in diesem Rahmen aber zwischen den farbigen Quadraten gefasst und gezogen, so lässt sich seine Position verändern.
   ACHTUNG: Unabhängig von allen Veränderungen an Geometrien ist es erforderlich, dass sich diese vollständig innerhalb der grauen Umrandung befindet, welche den Arbeitsbereich markiert. Dabei handelt es sich um den Bereich, den der Scanner maximal erreichen kann, Geometrien außerhalb dieses Bereiches können nicht markiert werden und würden abgeschnitten.
6. **Bestehende Geometrien manuell modifizieren**: Bei der im vorigen Schritt beschrieben Methode handelt es sich um einen schnellen aber nicht sehr genauen Weg, um bestehende Geometrien zu verändern. Aus diesem Grund existiert noch eine zweite, deutlich genauere Methode, bei der die benötigten Werte von Hand eingegeben werden können. So lange ein Element selektiert ist, sind auf der linken Seite des Hauptfensters verschiedene Tab-Panes aktiv und benutzbar. Die oberste Pane ist dabei immer Element-spezifisch und enthält Parameter, um dessen Darstellung zu verändern. Auf der gleichen Seite existiert auch eine Tab-Pane “Geometry”, in welcher grundlegende geometrische Eigenschaften dieses Elementes verändert werden können. Hier können Größe, Position, Rotation, Ausrichtung und anderes durch Eingabe der entsprechenden Werte sehr exakt angegeben werden. Die so übergebenen Daten werden immer durch Betätigung der Return-Taste oder bei Verlassen der entsprechenden Eingabefelder übernommen und auf das selektierte Element angewandt.
   Einige der dort angebotenen Eingabemöglichkeiten überlappen einander und werden abhängig von den neu eingegebenen Werten automatisch angepasst. D.h. wird beispielsweise der

Skalierungsfaktor eines Elementes verändert und die Eingabe mit Return abgeschlossen, so werden anschließend automatisch auch die Werte für Breite und Höhe des Elementes angepasst.

7. **Laserparameter zuweisen**: Die zum Markieren benötigten Laserparameter lassen sich über den Weg vordefinierter so genannter „Pens“ den Geometrien zuordnen. Ein solcher, bereits existierender Pen kann bei selektierter Geometrie über die Tab-Pane “Element” und deren Auswahlliste “Pen” zugeordnet werden. Wann immer dort ein neuer Pen ausgewählt wird, so wird dieser der Geometrie zugewiesen, diese wird anschließend in der Farbe des Pens dargestellt und während der Markieroperationen werden die Werte aus diesem Pen für diese Geometrie verwendet.
   Die verschiedenen Laser– und Scannerparameter, die zu so einem Pen gehören, können über das Menü “Project”, Menüpunkt “Pen settings...” geändert werden. Hier öffnet sich ein Dialog, in welchem der zu ändernde Pen ausgewählt werden kann (über die Auswahlliste am oberen Rand des Dialogfensters). Die Tab-Panes direkt darunter bieten Zugriff auf die diversen Parameter, hier können Markier- und Sprunggeschwindigkeit, Laserleistung, Frequenz, Scanner-Delays und andere Werte mehr verändert werden. Werden hier Werte geändert und wird der Dialog anschließend mit “OK” verlassen, so werden bei der nächsten Markieroperation all diejenigen Geometrien mit den geänderten Werten abgearbeitet, denen der soeben geänderte Pen zugeordnet ist.

8. **Ein Schraffurmuster hinzufügen**: Das aktuell bearbeitete Projekt enthält derzeit nur ein Dreieck, dessen Kontur während eines Markierprozesses gezeichnet werden würde. Um auch dessen Inneres zu markieren, muss es mit einem Schraffurmuster gefüllt werden. Das geschieht mit Hilfe des „Additional Geometry“-Elementes „Hatch“, in der Toolbar durch ein violettes Symbol mit horizontalen Linien symbolisiert. Wird dieses Symbol angeklickt, während ein Element vom Typ “Primary Geometry” selektiert ist (wie es das Dreieck eines ist), so werden die Hatch-Geometrien automatisch diesem Element zugeordnet, das Aussehen der Geometrie und der Applikation verändern sich:
   - die erste Tab-Pane auf der linken Seite des Hauptfensters schaltet um und zeigt jetzt statt der Parameter des zuvor gezeichneten und selektierten Dreiecks die “Hatch”-Parameter
   - der Elementtree auf der rechten Seite des Fensters zeigt die neue hierarchische Struktur des aktuellen Projektes mit dem “Hatch” als Unterelement des zuvor erzeugten Dreiecks
   - das Innere des Dreiecks im Zeichenbereich ist jetzt mit zusätzlichen Linien gefüllt, deren Aussehen sich über die Tab-Pane auf der linken Fensterseite ändern lassen
   ACHTUNG: Nach der Betätigung des Hatch-Buttons in der Toolbar ist jetzt die Schraffur im Inneren des Dreiecks selektiert, nicht aber das Dreieck selber! Das heißt, alle Änderungen an der Geometrie beziehen sich jetzt ausschließlich auf die Schraffur, das Dreieck selbst bleibt davon unberührt, so lange es selbst nicht wieder selektiert wurde!

9. **Daten abspeichern**: Um zu verhindern, dass die bisher erzeugten Daten verloren gehen, soll das aktuelle Projekt gespeichert werden. Dazu ist im Menü “Project” der Menüpunkt “Save as...” auszuwählen um das Projekt im BEAMP-Format an einer geeigneten Stelle abzuspeichern.
   ACHTUNG: wird beim Speichern angegeben, dass alle Projektdaten gesichert werden sollen, so enthält diese Projektdatei auch die zuvor vorgenommenen Hardwarekonfigurationen. Wird diese Konfiguration geändert und dann dieses Projekt wieder geladen, so werden die (alten) Hardwareeinstellungen aus dem Projekt aktiv. Um das zu verhindern bzw. um die neueren Projektparameter zurückzuerhalten ist nach dem Laden eines solchen Projektes im Menü “Project” der Menüpunkt “Load default configuration” auszuwählen. Alternativ kann beim Speichern auch angegeben werden, dass ein Projekt ohne die zugehörigen Hardwareeinstellungen (d.h. nur mit Geometrien und Pens) gespeichert wird.

10. **SICHERHEITSCHECK**: Im nächsten Schritt wird der Scannercontroller erstmalig angesteuert. Das heißt, er wird initialisiert und gestartet, angeschlossenes Equipment wird dann möglicherweise ebenfalls gestartet. Aus diesem Grund ist es wichtig, dass alle notwendigen Sicherheitsmaßnahmen getroffen wurden und die technischen Sicherheitseinrichtungen installiert und aktiv sind, um auch im Fall von Fehlverhalten oder spontaner Aktivität des Lasers oder anderer Komponenten keinen Schaden anzurichten und keine Personen zu verletzen!

11. **Vorbereitung zum Lasermarkieren**: Im nun folgenden Schritt sollen die bisher erzeugten Geometrien zum Laser und zum Scannersystem gesendet werden um das dafür vorgesehene Material entsprechend zu bearbeiten. Dazu ist im Menü “Process” der Menüpunkt “Mark” auszuwählen. Jetzt wird versucht, die angeschlossene Hardware zu initialisieren. Falls es der Software nicht möglich ist, auf die Scannerkarte zuzugreifen, wird eine Fehlermeldung angezeigt und der Markierdialog wird nicht geöffnet. In diesem Fall ist es erforderlich, die Konfiguration der angeschlossenen Hardware zu korrigieren (wie in Schritt 2 beschrieben) und den in der

Fehlermeldung angegebenen Fehler zu beheben.

12. **SICHERHEITSCHECK**: Im nächsten Schritt werden Laser und Scanner angesteuert. Nach dem es einige (Laser)Controller gibt, die SEHR empfindlich auf falsche Signale reagieren, empfiehlt es sich, diesen Schritt erstmalig ohne angeschlossene Hardware zu machen und die ausgegebenen Signale zu überprüfen (z.B. mit einem Oszilloskop). Das ist erforderlich, um Fehlkonfigurationen oder sonstige Probleme vorab zu identifizieren. Das sollte vor jeder Erstinbetriebnahme und vor jeder Inbetriebnahme nach Softwareupdates gemacht werden.
    „Softwareupdate“ umfasst hierbei nicht nur Veränderungen im Softwarestand des OpenAPC-Paketes sondern auch davon abhängende Software wie z.B. Firmware oder Treiber von Scannerkarten.
13. **Markieren starten**: im jetzt geöffneten Markierdialog kann der Markiervorgang durch Betätigung des großen Buttons mit dem gelben Laser-Warnschild darauf gestartet werden (zu finden in der Mitte des Markierdialoges). Dadurch werden alle Laser- und Scannerdaten zum angeschlossenen Scannercontroller gesendet, so dass dieser Spiegelbewegungen und Laseransteuerung synchron vornehmen kann.
    So ein Markiervorgang kann jederzeit durch Betätigung des großen roten Button mit dem Stopp-Symbol auf der rechten Seite des Dialoges abgebrochen werden.
    ACHTUNG: Dieser Stopp-Button ist kein Ersatz für einen echten Notstopp! Dieser versucht einen laufenden Markierprozess nur über die Software zu stoppen was – im schlimmsten anzunehmenden Fall – auch fehlschlagen kann. Aus diesem Grund muss jede Laseranlage einen davon unabhängigen, rein hardwaregesteuerten Not-Aus besitzen!

7.5 Projektkonfiguration

Wenn ein neues Projekt angelegt wird, so muss dieses zuerst so eingestellt werden, dass die Softwareumgebung den realen Gegebenheiten der Hardware entspricht. Das kann über die Projektkonfiguration im Menü “Projekt”, Untermenü “Projekteinstellungen...” erledigt werden. Wird dieser Menüpunkt ausgewählt, so öffnet sich ein Dialog mit mehreren Tab-Panes, von denen die wichtigsten hier beschrieben werden sollen.

Die Tabbed-Pane “Zeichenbereich”:

- Raster – hierbei handelt es sich um ein Hilfsraster, welches im Zeichenbereich als Orientierungs- und Zeichenhilfe angezeigt wird, dabei handelt es sich um ein Punktraster mit einem hier für X und Y anzugebenden Punktabstand
- An Raster ausrichten – wenn diese Checkbox selektiert ist werden alle Geometrien automatisch an dem oben konfigurierten Raster ausgerichtet. Sämtliche Anfangs-, End- und Stützpunkte befinden sich dann immer exakt an der Position eines Rasterpunktes. Diese Funktion bietet die Möglichkeit Geometrien sehr genau an diesen Fixpunkten auszurichten
- Hintergrundfarbe – diese Option beeinflusst nur die Darstellung des Zeichenbereiches, bewirkt aber keine Änderungen an den erzeugten Daten. Hiermit ist es möglich, die Hintergrundfarbe zu ändern
- Rahmen des Arbeitsbereiches: ähnlich zur vorhergehenden Option ist es hiermit möglich, die Farbe des Rahmens um den Arbeitsbereich herum zu verändern. Auch diese Option hat keinen Einfluss auf die erzeugten Markierdaten und damit auf das Markierergebnis
- Vergrößern / Elementtree in Tabbed Pane verschieben – mit dieser Option lässt sich Fensterplatz in horizontaler Richtung sparen. Ist die Checkbox gesetzt, wird der Elementbaum beim nächsten Programmstart nicht mehr auf der rechten Fensterseite, sondern als neues Tab auf der linken Seite angezeigt. Somit wird der Zeichenbereich in X-Richtung vergrößert.
- Tastaturabkürzungen – der Bereich unterhalb all dieser Optionen enthält einen Editor, mit dem es möglich ist, einzelnen Menüpunkten Tastaturkürzel zuzuordnen. Der gewünschte Menüpunkt kann dabei in der Baumansicht auf der unteren linken Seite dieser Tab-Pane ausgewählt werden. Anschließend ist es rechts im Feld “Neues Tastaturkürzel” möglich, die Tastenkombination einzugeben, welche diesem Menüpunkt zugeordnet werden soll und diese mit dem Button “Hinzufügen” festzulegen. Bereits bestehende Kürzel werden in der Liste darüber angezeigt und können mit dem Button “Entfernen” wieder gelöscht werden. Wird der Dialog zur Projektkonfiguration anschließend mit “OK” verlassen, so werden diese Abkürzungen auf die entsprechenden Menüpunkte angewendet und sofort wirksam.
  Diese Einstellungen sind dabei nur lokal wirksam und werden nicht Teil eines Projektes.

In der Tab-Pane “Benutzer” kann der Zugriff auf Softwarefunktionalitäten durch die Funktionen der Benutzerrechteverwaltung limitiert werden. Durch Setzen der Checkbox “Benutzerrechteverwaltung aktivieren” wird diese Verwaltung eingeschaltet und es werden verschiedene vordefinierte Einstellungen angewendet. Für alle auf diese Art gesetzten Benutzer ist das Passwort gleich dem Benutzernamen, folglich sollten diese vor einer Benutzung auf dem Zielsystem unbedingt geändert werden.
Diese Tab-Pane listet alle existierenden Benutzer auf der linken Fensterseite auf. Wird einer dieser Einträge ausgewählt, so ist es auf der rechten Seite möglich, die Rechte dieses Users, seinen Loginnamen, den vollen Namen, seinen Status, den zugehörigen Kommentar und das Passwort zu verändern. Um diese Änderungen wirksam werden zu lassen, muss anschließend der Button “Anwenden” gedrückt werden, Nur dadurch werden die modifizierten Werte auch tatsächlich übernommen.

Die Checkboxen auf der oberen rechten Seite dieser Tab-Pane legen fest welche Rechte ein zugehöriger Benutzer hat, nachdem sich dieser erfolgreich bei der Software angemeldet hat:

- Oberaufsicht – hierbei handelt es sich um Spezialprivileg, welches automatisch immer dem höchstprivilegierten Benutzer zugeordnet ist und welches vollen und uneingeschränkten Zugriff auf alle Funktionen der Software gewährt
- Benutzer verwalten – ein Benutzerprivileg, welches den Zugriff auf die Funktionen dieser Tab-Pane gewährt und es damit ermöglicht, die Benutzer und ihre Rechte zu verändern

- Applikation beenden – mit diesem Privileg ist ein Benutzer in der Lage die Software zu verlassen
- Einstellungen verändern – hiermit kann ein Benutzer die Projektkonfiguration öffnen und die globalen Einstellungen der Software verändern
- Geometrie bearbeiten – wenn dieses Privileg für einen Benutzer gesetzt ist kann dieser die Elemente eines geladenen Projektes verändern
- Projekt laden – dieses Privileg erlaubt es einem Benutzer ein anderes Projekt zu laden und entspricht damit dem aus anderer Software möglicherweise bekannten “Rezept laden”
- Projekt speichern – ein Benutzer mit diesem Privileg ist in der Lage das aktuelle Projekt abzuspeichern. Dieses Privileg sollte sinnvollerweise in Kombination mit der Möglichkeit das Projekt zuvor auch zu verändern eingesetzt werden (z.B. Geometrie bearbeiten, Projekt (dazu)laden, Projekteinstellungen bearbeiten)
- Markieren – wenn dieses Privileg für einen Benutzer gesetzt ist kann dieser den Markierprozess steuern

Im Unterschied zu den anderen Einstellmöglichkeiten des Konfigurationsdialoges werden diese Benutzerdaten nicht in den Projektdaten gespeichert, sondern ausschließlich lokal abgelegt. Das heißt also, dass eine Projektdatei, die auf ein anderes System transferiert wird, auf dem keine Benutzerverwaltung aktiv ist, völlig frei und ohne Einschränkungen verwendet werden kann.
Die Benutzerdaten selber werden im Standardverzeichnis für Applikationsdaten innerhalb eines Unterverzeichnisses “beamhome” abgelegt. Die exakte Position dieses Verzeichnisses hängt dabei vom verwendeten Betriebssystem ab. Die darin abgelegten Daten sind weder verschlüsselt noch sonstwie geschützt, so dass die Benutzerverwaltung deaktiviert ist, so bald dieses Verzeichnis gelöscht wird. Aus diesem Grund ist ein Schutz vor unbefugtem Zugriff mit den Mitteln des Betriebssystems erforderlich.
Die Möglichkeit das “beamhome”-Verzeichnis auf diese Art zu löschen erlaubt es berechtigten Usern im Falle des Verlustes wichtiger Benutzerpasswörter die Rechteverwaltung unkompliziert zurückzusetzen und neu zu konfigurieren.

Wurde die Benutzerrechteverwaltung aktiviert, so sind beim nächsten Programmstart alle per Privileg gesicherten Funktionen deaktiviert und werden erst dann benutzbar, wenn sich ein Benutzer erfolgreich mit Benutzernamen und korrektem Passwort anmeldet. Das geschieht über den Toolbarbutton mit dem Benutzersymbol:

Die Tabpane "Verschiedenes" enthält Optionen und Konfigurationsmöglichkeiten, welche nicht zu den vorhergehenden Kategorien passen:

- Aktiviere IOSelect-Modus – mit dieser Checkbox wird die IOSelect-Funktion (siehe Abschnitt unten) eingeschaltet und alle zugehörigen Einstellmöglichkeiten aktiviert
- Verzeichnis nummerierter Projekte – hier muss der Pfad zu einem Verzeichnis angegeben werden, welches die Projektdaten enthält, die abhängig vom Zustand eines Digitaleinganges geladen werden sollen
- Eingangs-Port an Karte 1 – der Digitaleingang, der für die externe Auswahl eines Projektes verwendet werden soll; dieser Parameter ist für den Fall wichtig, dass eine Karte mehr als eine digitale Eingangsschnittstelle bietet
- Markieren automatisch starten – ist diese Option gesetzt, so wird der Markiervorgang von automatisch geladenen Projekten sofort nach Öffnen des Markierdialoges gestartet. Achtung: diese Funktion kann gefährlich sein und sollte nur mit Bedacht eingesetzt werden! Der Start einer Markieroperation direkt nach dem Öffnen des Markierdialoges und ohne weitere Benutzerinteraktion kann für den Anwender unerwartet und überraschend sein!

7.6 Stifteinstellungen

Jedem in BeamConstruct erzeugten Element ist ein Stift zugeordnet. Dieser Stift enthält alle Informationen,

die für den Markierprozess der zugehörigen Geometrien bzw. der Bitmapdaten erforderlich sind: Scannergeschwindigkeiten, Delays, Laserleistung/-frequenz und anderes mehr. Diese Werte können im Dialog für die Stifteinstellungen verändert werden, welcher über das Menü “Projekt”, Menüpunkt “Stifteinstellungen...” erreicht werden kann.

Am oberen Rand dieses Dialoges kann der Kopf ausgewählt werden, für den die darunter gewählten Stifteinstellungen gelten sollen. Dies ist für Multihead-Umgebungen wichtig, in denen mehrere Scannercontroller und mehrere Arbeitsbereiche parallel benutzt werden und in denen der gleiche Stift abhängig vom Kopf andere Werte haben kann.

7.6.1 Allgemeine Stifteinstellungen

Dieses Panel enthält grundlegende Penparameter. Am oberen Rand befindet sich ein Eingabefeld, mit dem der Name des Stiftes verändert werden kann, Änderungen werden durch Betätigung der Eingabetaste aktiv. Darunter befinden sich verschiedene Regler und Eingabefelder für weitere Stiftparameter.

7.6.1.1 Stiftparameter-Wizard

Für Lasertypen, welche als Parameter eine Kombination aus Leistung und Frequenz benötigen, ist es – abhängig vom gewählten Material – manchmal schwierig, die passenden Werte zu finden. Darum bieten die Stifteinstellungen einen Wizard, welcher eine Matrix aus Markierungen unterschiedlicher Leistung und Frequenz erstellt, aus denen die richtigen Ergebnisse ausgewählt werden können. Um diesen Wizard zu öffnen ist der Button “Parameterwizard” direkt unterhalb der Eingabefelder für Leistung und Frequenz zu betätigen.
Dadurch öffnet sich ein neues Fenster mit einer 10x10 Felder Matrix von kleinen Markierbereichen, welche durch ebensolche Buttons symbolisiert werden. Im oberen Bereich des Fensters ist für diese Matrix ein Wertebereich für Leistung und Frequenz anzugeben, welcher über die Matrix verteilt wird. Als nächstes wird der gelbe “Start”-Button rechts neben der Matrix betätigt, um den Markierprozess zu starten. Anschließend wird das beste Markierergebnis ausgewählt und der zugehörige Button betätigt.

ACHTUNG: Durch Betätigung des Markier-Buttons wird eine Laseroperation gestartet! Diese kann mit einer Leistungsabgabe von bis zu 100% verbunden sein und ist deswegen potentiell gefährlich. Aus diesem Grund müssen alle Schutzmaßnahmen eingehalten werden und aktiv sein, bevor dieser Button betätigt wird!

Um das Auffinden des gewünschten Markierergebnisses zu erleichtern ist ein Markierelement als Kreis und eines als Dreieck ausgeführt. Damit kann die Orientierung und Lage der Matrix eindeutig identifiziert und die Zuordnung von Markierergebnis und Button per einfacher Zähloperation gefunden werden: Es werden erst die Schritte vom Dreieck bis zum Kreis gezählt, bis die richtige Spaltennummer aufgefunden wurde. Anschließend wird nur die Anzahl der Zeilen bis zum gewünschten Ergebnis gezählt. Wird die gleiche Zählreihenfolge auf die Buttonmatrix im Wizard angewendet, so lässt sich der richtige Button schnell auffinden. Dieser wird betätigt und aus dem sich öffnenden Popup-Menü der gewünschte Eintrag gewählt:

- Ergebnis verfeinern – die Matrix wird mit neuen Werten für Frequenz und Leistung initialisiert, wobei diese Werte sich näher am gewünschten Ergebnis befinden. Eine erneute Markieroperation wird also mit deutlich genaueren Werten ausgeführt
- Von diesem Ergebnis zurückgehen – diese Option verursacht die gegenteilige Wirkung. Hier wird die Matrix mit einem größeren Wertebereich und damit mit weniger feinen Abstufungen für Leistung/Frequenz reinitialisiert
- Parameter für Stift verwenden – wenn dieser Menüpunkt ausgewählt wird wird der Wizard geschlossen und die Frequenz- und Leistungswerte dieser Matrixposition werden in den aktuellen Stift übernommen.

Der Stiftparameter-Wizard benutzt für seine Funktion immer die Geschwindigkeits- und Delay-Einstellungen des aktuellen Stiftes, lediglich Frequenz und Leistung werden variiert.

7.7 IOSelect-Modus

Normalerweise werden Projektdateien manuell über die Auswahl des entsprechenden Menüpunktes unter Vorgabe des Pfades und Dateinamens geladen. Das kann mit Hilfe des IOSelect-Modus automatisiert werden. Ist dieser Modus aktiv, so wird der Zustand eines Digitalports einer angeschlossenen Scannerkarte gelesen und dessen Wert zum automatischen Laden einer Projektdatei verwendet. Ist dann der Markierdialog geöffnet und der Zustand am überwachten Digitalport ändert sich, so wird ein neues Projekt geladen und kann mit der nächsten Markieroperation ausgegeben werden. Ändert sich der Zustand der Eingänge während gerade eine Markieroperation läuft, so wird das Projekt geladen, so bald der aktuelle Markierzyklus beendet ist.

Um den IOSelect-Modus einzurichten sind folgende Schritte notwendig:

1. Erzeugen eines Verzeichnisses, in welchem die zu ladenden Projektdateien abgelegt werden
2. Erzeugen der zu ladenden Projektdateien, bei diesen muss der Dateiname aus einer Nummer und der Endung ".beamp" bestehen. Die Nummer korrespondiert dabei mit dem Bitmuster an der Digitalschnittstelle. Wenn dort z.B. das Bit 0 gesetzt ist, so lautet der Dateiname "1.beamp". Sind alle Bits eines 8 bit Digitalports gesetzt, so muss die zugehörige Datei "255.beamp" heißen. Sind keine Bits gesetzt, wäre der Dateiname "0.beamp", allerdings wird empfohlen, diesen Dateinamen nicht einzusetzen und 0 als neutralen Zustand ohne geladenes Projekt zu verwenden. Die so erzeugten Dateien müssen in dem im vorigen Schritt erzeugten Verzeichnis abgelegt werden.
3. In den Projekteinstellungen ist im Panel "Verschiedenes" der IOSelect-Modus zu aktivieren, das oben erzeugte Verzeichnis anzugeben und der zu verwendende Digitalport festzulegen. Dafür ist es notwendig, zuvor eine Scannerkarte vollständig zu konfigurieren.
4. Zum Aktiveren des automatischen IOSelect-Modus das Menü "Prozess" -> "Markieren" selektieren, so dass sich der Markierdialog öffnet

Folgendes ist beim IOSelect-Modus zu beachten:

- steht am Digitalport ein Bitmuster an, zu dem es keine passende Projektdatei gibt, so wird das aktuelle Projekt verworfen, so dass keine Markierdaten zur Verfügung stehen
- wird ein Projekt auf diesem Weg automatisch geladen, so werden die aktuellen Hardwareeinstellungen nicht überschrieben, d.h. alle Einstellungen des geladenen Projektes werden ignoriert und statt dessen die aktuellen Werte weiter verwendet
- im IOSelect-Modus wird der externe Trigger nicht automatisch aktiviert, um den Markierstart auf diesem Wege zu steuern. Soll dies geschehen, so müssen alle zu ladenden Projekte an erster Stelle mit einem External Trigger Primary Element versehen werden und im Markierdialog muss die Checkbox für den Markierloop gesetzt werden
- in Multihead-Umgebungen kann derzeit immer nur die erste Karte verwendet werden, um mit deren Digitaleingängen ein Projekt auszuwählen
- - der IOSelect-Modus wird erst gestartet, wenn der Markierdialog einmal geöffnet wird; wärend des Wechsels zu einem anderen Projekt wird dieser Dialog geschlossen und anschließend automatisch wieder geöffnet

7.8 BeamConstruct modifizieren

BeamConstruct kann auf regulärem Wege so verändert werden, dass es zusammen mit Maschinen oder zusätzlichem Equipment unter einem komplett eigenen Namen und Aussehen ausgeliefert werden kann. Folgende Eigenschaften können modifiziert werden:

- Der Name der Software

- Die Homepage im “About”-Dialog
- Das Logo, das beim Programmstart und im “About”-Dialog angezeigt wird
- Das in der Taskbar angezeigt Icon der Applikation (wird nicht auf allen Betriebssystemen unterstützt)
- Die Copyright-Information im “About”-Dialog; diese werden zu den bestehenden Copyright-Daten hinzugefügt
- Die Bilder der Toolbar und des Elementtrees

Um die Software auf diese Art zu modifizieren, wird eine einfache Textdatei im UTF-8-Format mit dem Namen `beam.oem` benötigt, welche im Installationsverzeichnis der Software abgelegt werden muss (für Windows typischerweise C:\Program Files\OpenAPC, auf Linux-Systemen in /usr/local/openapc). In dieser Datei enthält jede Zeile einen festen Wert:

6. Der Name der Software
7. Der URL der im “About”-Dialog anzuzeigenden Homepage
8. Pfad zu einer Bilddatei im PNG-Format, welche als Logo angezeigt werden soll
9. Pfad zu einer kleinen Bilddatei im PNG-Format, welche als Icon angezeigt werden soll
10. Copyrighttext, der im “About”-Dialog angezeigt werden soll

Der Inhalt dieser Datei kann beispielsweise so aussehen:

```
Beamy Raymanipulator
http://www.beamconstruct.com
beamoem.png
beamoem_icon.png
Copyright (c) 2013 by LaserFreak
```

Wird diese Datei (als UTF-8–codierte Textdatei) im oben beschriebenen Verzeichnis abgelegt, so wird “BeamConstruct” zu “Beamy Raymanipulator” umbenannt. Die Software zeigt dann im “About”-Dialog den URL http://www.beamconstruct.com (anstatt http://www.lasermarkingsoftware.com) sowie die zusätzliche Copyright-Information “Copyright (c) 2013 by LaserFreak ”. Als Logo für den Startup-Screen und den “About”-Dialog wird das Bild aus beamoem.png verwendet. Wird dieser Dateiname ohne Verwendung eines absoluten Pfades angegeben, so versucht die Software, das Bild im Installationsverzeichnis (Windows) oder in /usr/local/openapc (Linux) zu finden. Gleiches gilt für die Datei beamoem_icon.png, welche in diesem Beispiel das Icon für die Applikation bereitstellt.

Die Bilder in der Toolbar sowie im Elementtree können ebenfalls ausgetauscht werden, Hier kommen zwei unterschiedliche Methoden zum Einsatz.

Um die Toolbar-Symbole zu ersetzen werden für jedes Plug-In zwei Bilder im PNG-Format benötigt, welche in den Installationsverzeichnissen dieser Plug-Ins abgelegt werden müssen. Der Basisname der Bilder ist dabei jeweils identisch mit dem Plug-In-Namen (ohne Dateinamenserweiterung .DLL oder .so). Für das Toolbar-Symbol muss dem Dateinamen ein “1.png” hinten angestellt werden, für das kleinere, zugehörige Symbol im Elementtree entsprechend ein “2.png”.
Als Beispiel: Die Symbole des Barcode-Plug-Ins sollen ersetzt werden. Dieses kann im Ordner “priplugins” unter dem Namen “libio_pri_barcode.dll” bzw. “libio_pri_barcode.so” gefunden werden. Es werden an dieser Stelle also ein Bild “libio_pri_barcode1.png” mit einer Größe von 25x25 Pixeln und ein Bild “libio_pri_barcode2.png” mit einer Größe von 15x15 Pixeln benötigt.

Um alle anderen Bilder, welche nicht direkt zu einem Plug-In gehören, zu ersetzen müssen PNG-Dateien unter folgenden Namen im Unterordner “icons” abgelegt werden:

- “newprj.png” für das Symbol “Neu“
- “openprj.png” für das Symbol “Projekt laden”
- “saveprj.png” für das Symbol “Projekt speichern”
- “saveprjas.png” für das Symbol “Projekt speichern als”

- “quit.png” für das Symbol “Applikation beenden”
- “zoomin.png” für das Symbol “Hereinzoomen”
- “zoomout.png” für das Symbol “Herauszoomen”
- “zoomworkarea.png” für das Symbol “Auf Arbeitsbereich zoomen”
- “zoomselected.png” für das Symbol “Auf selektiertes Element zoomen”
- “pointdrag.png” für das Symbol “Punktmodus”

7.9 BeamLock

BeamLock ist ein Hilfsmittel, mit dem eigene Sperrichtlinien auf einem System definiert werden können. Das kann z.B. für Maschinenintegratoren nützlich sein, wenn Hardware zusammen mit einer BeamConstruct-Installation ausgeliefert wird, bevor der Kunde eine Rechnung vollständig bezahlt hat. In so einem Fall kann eine Sperre auf diesem System gesetzt werden, welche nach einer konfigurierbaren Zeit aktiv wird. Das bedeutet, dass die Funktionen dieser Maschine nur zeitlich begrenzt zur Verfügung stehen. Der Benutzer muss nach Ablauf dieser Zeitspanne ein Passwort eingeben, welches vom Hersteller dieser Maschine nur dann zur Verfügung gestellt wird, wenn eine Rechnung bezahlt ist bzw. Lizenzbestimmungen oder Vereinbarungen eingehalten wurden. Ohne dieses Passwort ist die Softwareinstallation nicht länger verwendbar. Auch eine Deinstallation und erneute Installation von BeamConstruct kann diese Sperre nicht umgehen.

BeamLock bietet die Möglichkeit bis zu drei Sperrzeiträume und zugehörige Passwörter festzulegen. Damit ist es möglich mehrere Sperrstufen zu realisieren, welche nacheinander und jeweils nach Ablauf der gesetzten Zeitspanne aktiv werden. Zusätzlich kann ein Masterpasswort definiert werden, mit dem sämtliche zeitgesteuerten Sperren auf einmal deaktiviert werden können.

ACHTUNG: Alle Passwörter inklusive des Masterpasswortes sollten sich aus Sicherheitsgründen deutlich unterscheiden. Auch ist es empfehlenswert, komplexe Kombinationen aus Groß-, Kleinbuchstaben und Zahlen zu verwenden, welche keine sinnvollen Wörter enthalten. Beispiele für SCHLECHTE PASSWÖRTER sind: "sperre1", "sperre2", "sperre3", "sperrem". Beispiele für gute Passwörter sind hingegen: "XdlfTr238", "yERf12rt67", "934trAwX4", "m4RT56ZZtgd34".

Um eine solche Sperre auf einem System einzurichten sind folgende Schritte notwendig:

1) BeamLock auf dem zu sperrenden System starten: Unter Windows kann die BeamLock.exe im OpenAPC-Installationsverzeichnis gefunden werden. Unter Linux ist "BeamLock" in der Konsole einzugeben. Auf keinem der unterstützten Betriebssysteme wird BeamLock im Startmenü angezeigt.
2) Setzen der Checkbox "Enable" in jeder Zeile, in der ein Zeitraum mit zugehörigem Entsperrpasswort definiert werden soll. Hier muss jeweils auch die Zeit, nach der die Sperre aktiv werden soll zusammen mit dem Passwort eingegeben werden.
3) Setzen eines Masterpasswortes.
4) Aktivieren des Locks durch Betätigen des Buttons "Lock System". Wird das auf einem bereits gesperrten System gemacht, muss zum Überschreiben der aktuellen Sperre das derzeitige Masterpasswort eingegeben werden. Wenn ein System zuvor nicht bereits gesperrt war und eine neue Installation enthält, die Software aber dennoch nach einem Entsperrpasswort fragt, so ist "HALaser" als Passwort einzugeben. Sollte "HALaser" (ohne die Anführungszeichen) als Entsperrpasswort beim Anlegen einer neuen Sperre nicht funktionieren, ist das System bereits mit einer Sperre versehen worden. Diese kann ohne Kenntnis des Masterpassworts der bereits vorhandenen Sperre nicht überschrieben werden. Es existiert kein Weg, um eine solche Sperre ohne Kenntnis des Masterpasswortes zu entfernen.
5) Passwörter notieren und sicher verwahren! ACHTUNG: Diese Passwörter und speziell das Masterpasswort sind der EINZIGE Weg um ein gesperrtes System nach Ablauf des eingegebenen Zeitraumes freizuschalten oder eine neue Sperre zu setzen! Noch einmal: Es existiert kein Weg, um eine solche Sperre ohne Kenntnis des Masterpasswortes zu entfernen, diese Passwörter müssen also unbedingt sicher aufbewahrt werden!

ACHTUNG: Unter Windows ist diese Sperrprozedur abhängig von den Benutzerrechten, mit denen BeamConstruct ausgeführt wird. Läuft die Installation voreingestellt mit den Rechten eines Administrator-Accounts, so müssen die oben beschriebenen Schritte mit BeamLock.exe einmal unter Verwendung dieses Administratoraccounts durchgeführt werden. Wird BeamConstruct unter Verwendung eines nichtprivilegierten Benutzeraccounts ausgeführt, müssen diese Schritte mit BeamLock.exe in jedem Fall zweimal ausgeführt werden: einmal mit diesem Benutzeraccount und einmal mit einem Administratoraccount! In diesem Fall ist es empfehlenswert, in beiden Fällen die gleichen Passwörter zu verwenden, um Konfusionen zu vermeiden, falls der Endbenutzer unerwartet Änderungen an der Installation vornimmt.

ACHTUNG: Für Linux wird angenommen, dass BeamConstruct nicht als root sondern als nichtprivilegierter Benutzer ausgeführt wird. In diesem Fall müssen die oben beschriebenen Schritte mit BeamLock in jedem Fall zweimal ausgeführt werden: einmal mit diesem Benutzeraccount und einmal mit dem root-Account! In diesem Fall ist es empfehlenswert, in beiden Fällen die gleichen Passwörter zu verwenden, um Konfusionen zu vermeiden, falls der Endbenutzer unerwartet Änderungen an der Installation vornimmt.

7.10 BeamServer Remote Control Interface

7.10.1 Überblick

BeamConstruct bietet einen Modus an, in welchem die gesamte Applikation ferngesteuert werden kann. Das ist mit Hilfe eines externen Skriptes oder Programmes möglich. Dabei kann die BeamConstruct-Oberfläche auch komplett verborgen werden, während im Vordergrund andere Operationen ausgeführt werden. Für diese Art der Fernsteuerung von BeamConstruct wird ein eigenes Programm angeboten: der BeamServer. Wird dieser an Stelle von BeamConstruct gestartet (BeamServer.exe unter Windows, BeamServer wenn Linux verwendet wird), so wird auf Port 11350 ein TCP-Serversocket erzeugt. Jetzt kann sich eine externe Applikation mit diesem Socket verbinden und spezielle ASCII-Kommandos senden. Diese Kommandos führen dann zur Verwendung und Ansteuerung von BeamConstruct.
Wichtig: BeamConstruct muss in diese Konstellation nicht separat gestartet werden, das wird vom BeamServer automatisch gemacht.

7.10.2 Verwendung

Eine Standardsequenz von BeamServer-Kommandos sieht immer folgendermaßen aus:

1. Starten des BeamServer mit einer Nummer als Kommandozeilenparameter, welcher festlegt, welche Teile des BeamConstruct-Userinterface gezeigt werden sollen
2. Herstellen einer TCP-Verbindung zu dem Computer, auf dem das OpenAPC-Softwarepaket installiert ist und auf dem der BeamServer läuft; die Verbindung muss zum Port 11350 hergestellt werden
3. und alle weiteren: senden verschiedener ASCII-Kommandos zur Ansteuerung von BeamConstruct (wie unten beschrieben)

abschließend: Senden des ASCII-Kommandos

```
ExitUI
```

um BeamConstruct und den BeamServer wieder zu beenden

Wird die TCP-Verbindung zum BeamServer innerhalb dieser Sequenz unterbrochen, so wird BeamConstruct NICHT geschlossen. Nach der Wiederherstellung der Verbindung zum BeamServer kann die ansteuernde Applikation also an der Stelle vor der Verbindungsunterbrechung fortsetzen. Unabhängig davon erlaubt der BeamServer immer nur eine eingehende Verbindung gleichzeitig.

Zu Testzwecken ist es ebenfalls möglich, sich mit einem normalen Telnet-Programm auf den BeamServer zu verbinden und manuell Kommandos abzusetzen. Für diesen Zweck muss der Tenlen-Client im RAW-Modus betrieben werden (auto-negotiation deaktiviert).

7.10.3 BeamServer starten

Der BeamServer muss mit einem Kommandozeilenparameter gestartet werden. Dabei handelt es sich um eine Nummer, welche festlegt, welche Teile der BeamConstruct-GUI angezeigt werden sollen. Wenn diese Nummer 0 ist, so arbeitet BeamConstruct komplett unsichtbar im Hintergrund. Wird 255 übergeben, so wird die komplette GUI angezeigt. Es können auch andere Zahlenwerte angegeben werden, welche sich aus der Addition der folgenden Werte ergeben und darüber festlegen, welche Teile der GUI angezeigt werden sollen:

1 – Zeichenbereich anzeigen (der große Bereich in der Mitte, in dem Vektordaten gezeichnet werden können)

2 – Toolbars anzeigen (am oberen Rand des Hauptfensters)

4 – Element-Panels auf der linken Fensterseite anzeigen

8 – Element-Baum auf der rechten Fensterseite anzeigen

16 – Menübar mit allen Menüpunkten anzeigen

32 – Schließen-Button im Rahmen des Fensters anzeigen

64 – Statusbar am unteren Fensterrand anzeigen

131072 - deaktiviert alle Möglichkeiten für den User, die Applikation zu beenden, wenn dieser Wert verwendet wird, dann ist es nur noch über das Kommando "ExitUI" möglich, BeamConstruct zu verlassen

262144 - deaktiviert alle Möglichkeiten, mit deren Hilfe ein Benutzer Elemente des aktuellen Projektes verändern kann; wenn dieser Wert gesetzt ist, so ist es auch nicht möglich, ein selektiertes Element zu verschieben, zu skalieren, zu rotieren oder zu neigen

524288 - die Warnmeldung beim Programmstart wird deaktiviert; dieser Wert sollte nur in den seltensten Fällen benutzt werden und auch nur dann, wenn der Benutzer über Art und Funktion der Software informiert ist, so dass es zu keinen überraschenden und daher gefährlichen Operationen kommen kann

1048576 - wirkt sich nur auf das Layout des Markierdialoges aus, wenn dieser Wert verwendet wird, so wird die manuelle Scannerkontrolle im Markierdialog nicht angezeigt

2097152 - wirkt sich nur auf den Markierdialog aus und versteckt dessen Teilezähler-Tabpane

4194304 - wirkt sich nur auf den Markierdialog aus, dieser Wert unterdrück die "Motion"-Tabpane

8388608 - wirkt sich nur auf den Markierdialog aus, wenn dieser Wert gesetzt wird, so wird die "Geometrie"-Tabpane im Markierdialog nicht angezeigt

16777216 - wirkt sich nur auf den Markierdialog aus und versteckt dort die Pilotlaser-Tabpane

33554432 - hat nur dann eine Wirkung, wenn der Wert 2 (siehe oben) gesetzt ist und wenn der Wert 1073741824 (siehe unten) nicht verwendet wird, diese Option verbirgt den Stiftparameter-Button in der "Element"-Tabpane

67108864 - damit wird der Zeichenbereich für dem Fall gesperrt, dass dieser über den Wert 1 sichtbar gemacht wurde; dadurch kann seine Position nicht mehr durch Ziehen mit der Maus verschoben werden; ein Verändern des Zeichenbereiches ist nur noch mit den Funktionen möglich, die den aktuellen Zoom verändern

134217728 - hat nur dann eine Wirkung, wenn der Wert 2 (siehe oben) gesetzt ist, dieser Wert unterdrückt die Möglichkeit, einzelne Punkte einer selektierten Geometrie zu editieren; wird dieser Wert gesetzt, so ist das Koordinaten-Grid in der Tabpane "Geometrie" deaktiviert; es ist nicht notwendig, diesen Wert zu verwenden, wenn bereits der Wert 262144 (siehe oben) zum Einsatz kommt

1073741824 - hat nur dann eine Wirkung, wenn der Wert 2 (siehe oben) gesetzt ist, wenn dieser Wert

verwendet wird, so wird die "Element"-Tabpane nicht angezeigt

536870912 - hat nur dann eine Wirkung, wenn der Wert 2 (siehe oben) gesetzt ist, damit wird die dynamische Tabpane deaktiviert, welche die Parameter des aktuell selektierten Elements anzeigt

1073741824 - hat nur dann eine Wirkung, wenn der Wert 2 (siehe oben) gesetzt ist, damit wird die Ebenen-Tabpane deaktiviert und versteckt

2147483648 - die lokalen Vorgabeeinstellungen werden nicht geladen, statt dessen startet die Software mit einer (leeren) Defaultkonfiguration, wenn dieser Wert gesetzt wird

Als Beispiel: "BeamServer.exe 17" würde BeamConstruct bestehend nur aus dem Zeichenbereich (=1) und der Menübar (=16) anzeigen

Für Windows ist zu beachten: Die BeamServer.exe muss aus dem OpenAPC Installationsverzeichnis heraus gestartet werden. Um sicher zu stellen, dass dieses Verzeichnis auch wirklich als Arbeitsverzeichnis verwendet wird, empfiehlt es sich, direkt von dort aus eine Verknüpfung zu erstellen. Diese beinhaltet dann automatisch den Pfad zum richtigen Verzeichnis. Auch ist es möglich, innerhalb der Verknüpfungseinstellungen den oben beschriebenen Parameter zu setzen. Wird BeamServer.exe mit irgend einem anderen Arbeitsverzeichnis gestartet, so funktioniert das Programm nicht korrekt und findet keine Elemente und Plug-Ins.

7.10.4 Remote Control Kommandos

Jedes zum BeamServer via TCP/IP gesendete Kommando wird entweder mit dem Text "OK" beatwortet (wenn die zugehörige Operation erfolgreich ausgeführt werden konnte), oder mit dem Antwortstring "ERROR" (wenn das letzte Kommando fehlgeschlagen ist). "ERROR" wird dabei immer von zwei Fehlercodes gefolgt wobei der erste Code die Art des Fehlers beschreibt und der zweite Fehlercode eine interne Fehlernummer darstellt, die bei Problemen mit dem BeamServer selbst hilfreich sein kann. Für den ersten Fehlercode sind die folgenden Werte möglich:

7 – OAPC_ERROR_RESOURCE – es wurde versucht, auf eine nicht existierende Ressource zuzugreifen (z.B. wenn eine UID oder der Name eines Elementes verwendet werden, welches im aktuellen Projekt nicht existiert)

11 – OAPC_ERROR_NO_MEMORY – es ist nicht genügend Speicher vorhanden

16 – OAPC_ERROR_INVALID_INPUT – die mit dem letzten Kommando übergebenen Daten oder Parameter sind unvollständig oder falsch

18 – OAPC_ERROR_CREATE_FILE_FAILED – es war nicht möglich, eine Datei neu zu erzeugen, die Datei konnte nicht zum Schreiben geöffnet werden

19 – OAPC_ERROR_OPEN_FILE_FAILED – es war nicht möglich, eine Datei zum Lesen zu öffnen (z.B. weil die Datei nicht existiert oder aus anderen Gründen nicht zugreifbar ist)

20 – OAPC_ERROR_WRITE_FILE_FAILED – es war nicht möglich, Daten in eine bereits geöffnete Datei zu schreiben (z.B. weil kein Speicherplatz mehr frei ist)

21 – OAPC_ERROR_READ_FILE_FAILED – es war nicht möglich, Daten aus einer bereits geöffneten Datei zu lesen (z.B. weil die Datei beschädigt ist oder in einem falschen bzw. unbekannten Format vorliegt)

Der BeamServer unterstützt folgende Kommandos:

`ExitUI`

Dieses Kommando stoppt BeamConstruct und - so fern sichtbar - schließt dessen GUI komplett. Ebenso wird der BeamServer und dessen TCP/IP Socket geschlossen. Anschließend ist es nicht mehr möglich,

irgend ein anderes Kommando zu senden.

CmdStopMark

Versucht, eine laufende Markieroperation auf die gleiche Art zu stoppen, wie es auch durch einen User bei Betätigung des "STOPP"-Buttons im Markierdialog gemacht werden würde.

TriggerUI <number>

Dieses Kommando kann verwendet werden, um Operationen zu starten, wie sie auch durch den Benutzer innerhalb von BeamConstruct durch Verwendung der GUI möglich wären. Der Wert `<number>` legt dabei fest, welche Operation getriggert werden soll. Die meisten dieser Kommandos kehren sofort zurück, unabhängig davon, ob die angestoßene Operation erfolgreich war oder nicht. Synchrone Kommandos, bei denen das nicht der Fall ist und bei denen die Antwort erst nach Beendigung der jeweiligen Aktion gesendet wird, sind unten explizit erwähnt. Durch die gegebene `<number>` ist jede Benutzeraktion wie das Selektieren eines Menüpunktes oder die Betätigung eines Toolbar-Buttons möglich. Es werden folgende <number>-Werte unterstützt:

Alle vierstelligen Werte korrespondieren mit Menüpunkten oder Toolbar-Buttons und deren Funktion innerhalb von BeamConstruct.

Detailliertere Informationen zu den jeweiligen Funktionen finden sich in der Beschreibung zu BeamConstruct weiter oben:

1001 – neues Projekt
1002 – lade existierndes Projekt
1003 – importiere anderes Dateiformat (wie von BeamConstruct unterstützt: Vektor- und Rasterimage-Formate)
1004 – speichere das aktuelle Projekt unter Verwendung des bekannten Dateinamens
1005 – speichere das aktuelle Projekt unter einem neuen Namen
1101 – exportiere Vektordaten im CSV-Format
1102 – exportiere Vektordaten im HPGL/PLT-Format
1103 – exportiere Ebenen-Vektordaten im CLI-Format
1006 – öffne den Dialog für die Projekteinstellungen
1007 – öffne den Dialog für die Penparameter
1008 – speichere die aktuelle Konfiguration als Vorgabeparameter
1009 – lade die Vorgabeparameter
1010 – speichere das aktuelle Projekt mit Optionen
2001 – letzte Aktion rückgängig machen
2002 – letzten rückgängig gemachten Schritt wiederherstellen
2101 – alle selektierten Elemente in eine Gruppe zusammenfügen
2102 – alle selektierten Elemente in eine Split-Gruppe zusammenfügen
2103 – alle selektierten Elemente in eine Move-Gruppe zusammenfügen
2104 – die Elemente innerhalb der selektierten Gruppe separieren
2105 – das selektierte Element duplizieren
2201 – die Geometrien des selektierten Elements in eine Statische Geometrie zusammenführen
2202 – die Geometrien des selektierten Elementes logisch aufsplitten
2203 – die Geometrien des selektierten Elementes optimieren
2204 – die Geometrien des selektierten Elementes unter verwendung eines verlustbehafteten Algorithmus reduzieren
2205 – das selektierte Element in die dritte Dimension tiefziehen
2301 – das aktuell angezeigte Hintergrund-Videobild einfrieren
2302 – eine angeschlossene Kamera kalibireren
2303 – Marken lernen
2304 – Marken laden
2305 – gelernte Marken speichern
2306 – gelernte Marken verwerfen
2307 – Kalibrierdaten der Kamera verwerfen
2308 – den Bildeinzug an- oder ausschalten (so fern in den Einstellungen als manuell zu Starten konfiguriert)

3001 – Simulation des Prozesses starten
3002 – öffnet den MArkierdialog, aus dem heraus der Benutzer den Markiervorgang steuern kann
3003 – markiere das aktuelle Projekt einmal, hierzu wird der Markierdialog geöffnet, so dass der Benutzer den Vorgang stoppen kann; der Dialog wird nach Ende des Markiervorganges automatisch geschlossen; die Antwort "OK" auf dieses Kommando wird erst gesendet, wenn dieser Vorgang beendet ist, es ist also nicht notwendig, den Markierstatus separat zu überwachen
3004 – markiere das aktuelle Projekt einmal im Hintergrund, der Markierdialog wird hier nicht geöffnet; die Antwort "OK" auf dieses Kommando wird erst gesendet, wenn dieser Vorgang beendet ist, es ist also nicht notwendig, den Markierstatus separat zu überwachen
3006 – inkrementiere die Elemente im aktuellen Projekt
3007 – dekrementiere die Elemente im aktuellen Projekt
3008 – setze die Elemente im aktuellen Projekt auf ihre Vorgabe-/Startwerte zurück
3011 – öffne die konfigurierten Geräte (Scannerkarten, Motioncontroller, Bildeinzug,...)
3012 – Schließe geöffnete Geräte
3013 – öffne Geräte (wenn nicht bereits geschehen) und zeige den Dialog mit dem Scannerkartenstatus
3020 - dieses Kommando verhält sich ähnlich wie 3003, es öffnet den Markierdialog, startet die Markieroperation und schließt den Dialog automatisch, wenn die Operation beendet wurde; die Rückgabeinformation "OK" für dieses Kommando wird erst gesendet, wenn der komplette Vorgang beendet wurde. Im Unterschied zum Kommando 3003 setzt dieses auch die "Markierloop"-Checkbox im Markierdialog so dass das aktuelle Projekt wiederholt markiert wird. Zusätzlich wird diese Checkbox deaktiviert, so dass der Benutzer die Loop-Option nicht ausschalten kann. Der Markierdialog kann in diesem Fall nur durch Betätigung des Stop-Buttons oder durch Anklicken des Schließen-Symbols des Fensters beendet werden.
4001 – zeige den "About"-Dialog
4002 – zeige Lizenzinformationen
4003 – setze die aktuelle Lizenz zurück
4004 – zeige den "Credits"-Dialog

Alle fünfstelligen Nummern gehören zu den verbleibenden Toolbar-Buttons:

10001 – zoom into view
10002 – zoom out of view
10003 – zoom to view full working area
10004 – zoom to fully view selected element
10005 – enable pointdrag mode for selected element
10006 – log in a user

`CmdNewPrj`

Löscht alle aktuellen Daten (Stifte, Einstellungen und Elemente). Anschließend werden die Vorgabeeinstellungen wiederhergestellt, das Projekt ist also leer und enthält keinerlei Geometrien mehr. Wird anschließend das Kommando "`CmdAppendPrj`" verwendet, um ein neues Projekt zu laden, so werden die aktuellen (Vorgabe-)Einstellungen beibehalten.

`CmdLoadPrj <path>`

Lädt eine neue .BEAMP Projektdatei vom der mittels `<path>` angegebenen Stelle. Die in dem Projekt enthaltenen Elemente werden zu den aktuell bereits geladenen hinzugefügt (sofern das Projekt bereits Geometrien enthält). Die aktuellen Einstellungen und Stifte werden mit denen des neu geladenen Projektes überschrieben.

`CmdAppendPrj <path>`

Lädt eine neue .BEAMP Projektdatei vom der mittels `<path>` angegebenen Stelle. Die in dem Projekt enthaltenen Elemente werden zu den aktuell bereits geladenen hinzugefügt (sofern das Projekt bereits Geometrien enthält). Die aktuellen Einstellungen und Stifte werden beibehalten, die entsprechenden Parameter in der neu geladenen Projektdateien werden ignoriert.

CmdSetCharIO2 <new text>
CmdSetCharIO3 <new text>
CmdSetCharIO4 <new text>
CmdSetCharIO5 <new text>

Setzt einen neuen Text, der für alle Eingabeelemente gültig ist, welche vom logischen "Char"-Eingang gebrauch machen. Wenn ein solches Eingabeelement einem Text- oder Barcodeelement zugeordnet ist, so wird dieser Text oder Barcode für die nächste Markieroperation durch die als Parameter `<new text>` übergebenen Daten ersetzt. Die vier Kommandos für "CharIO" 2..5 entsprechen den Eingängen 2..5 welche im jeweiligen Eingabeelement ausgewählt werden können.

CmdWriteMark <path>

Mit diesem Kommando ist es möglich, die aktuellen Projektdaten als Stand-Alone-Markierdaten in eine lokale Datei zu speichern. Das erfordert eine Konfiguration, bei der eine Scannerkarte verwendet wird, die dieses Feature unterstützt (wie die HALaser E1701 Controllerkarten). Der von diesem Kommando erwartete Parameter <path> legt dabei den kompletten Pfad zu der Datei fest, in der die Stand-Alone-Markierdaten gespeichert werden sollen. Sollte diese Datei bereits existieren, so wird sie ohne weitere Warnung überschrieben. Eine so erzeugte Datei kann anschließend manuell zur Scannerkarte übertragen werden (z.B. in dem Sie direkt auf die microSD-Karte der E1701 gespeichert wird, welche anschließend wieder in den Controller eingesetzt wird).

CmdSendMark

Mit diesem Kommando ist es möglich, die aktuellen Projektdaten als Stand-Alone-Markierdaten an eine angeschlossene Scannerkarte zu senden. Das erfordert eine Konfiguration, bei der eine Scannerkarte verwendet wird, die dieses Feature unterstützt (wie die Scanlab RTC4 ScanAlone). Hier wird keine weiterer Parameter benötigt, da diese Funktion nur Exklusivdaten an eine Karte sendet.

CmdSendNamedMark <name>

Mit diesem Kommando ist es möglich, die aktuellen Projektdaten unter Verwendung eines spezifischen Namens als Stand-Alone-Markierdaten an eine angeschlossene Scannerkarte zu senden. Das erfordert eine Konfiguration, bei der eine Scannerkarte verwendet wird, die dieses Feature unterstützt (wie die HALaser E1701 Controllerkarten). Der übergebene Parameter <name> identifiziert dabei die auf der Karte zu speichernden Markierdaten. Die Struktur des Namens muss dabei den von der Karte vorgegebenen Konventionen entsprechen. Für den E1701-Controller bedeutet das, der Name muss im Format "0:/filename.epr" vorliegen, wobei "0:/" die microSD-Karte als Speicherort festlegt und ".epr" die empfohlene Dateinamenserweiterung darstellt. Existiert bereits eine Datei unter dem angegebenen Namen auf dem Controller, so wird diese ohne weiteres Nachfragen überschrieben.

CmdSetSelEnt <0/1>

Selektiert oder deselektiert ein Element sichtbar im View. Diese Operation ist identisch mit einem Anklicken eines Elementes durch den Benutzer bei heruntergedrückter Shift-Taste. Ein angefügter Parameter 0 oder 1 gibt dabei an, ob das Element selektiert oder deselektiert werden soll. Das Element selber, dessen Selektions-Zustand durch diese Funktion verändert wird, kann zuvor durch einen Aufruf `CmdSelEntName` vorausgewählt werden (dieses Kommando setzt eine interne, nicht sichtbare Selektierung für das Element). Nach dem ein Element sichtbar selektiert wurde, können all die Operationen ausgeführt werden, die sich auf selektierte Elemente beziehen. Beispielsweise ist für das Zoomen auf ein selektiertes Element folgendes auszuführen:

```
CmdSelEntName elementname
CmdSetSelEnt 1
CmdZoomElem
```

Um den Zoom auf zwei Elemente zu erweitern, kann diese Kommandosequenz verwendet werden:

```
CmdSelEntName elementname1
```

```
CmdSetSelEnt 1
CmdSelEntName elementname2
CmdSetSelEnt 1
CmdZoomElem
```

Um ein zuvor selektiertes Element wieder zu deselektieren, sind diese Schritte erforderlich:

```
CmdSelEntName elementname1
CmdSetSelEnt 0
```

CmdZoomElem

Zoomt den View so, dass alle selektierten Elemente möglichst groß und vollständig angezeigt werden. Die Elemente, die nach dieser Operation auf diese Art sichtbar sein sollen, müssen zuvor per "`CmdSelEntName`" und "`CmdSetSelEnt`" selektiert werden.

CmdRefresh

Frischt den aktuellen View auf. Dieses Kommando aktualisiert die angezeigten Geometrien ohne dabei eine Neuberechnung auszuführen.

CmdRedraw

Zeichnet den aktuellen View neu. Dabei wird eine vollständige Neuberechnung der angezeigten Geometrien ausgeführt, weswegen dieses Kommando langsamer als "`CmdRefresh`" ist. Deswegen sollte es nur dann aufgerufen werden, wenn im aktuellen Projekt Veränderungen an den Geometrien vorgenommen wurden, welche noch nicht sichtbar sind.

CmdListName <num>

Dieses Kommando liefert die Namen aller in einem Projekt vorhandenen Elemente zurück. Es erwartet eine positive Zahl `num` als Parameter, welche als Zähler für die aufzulistenden Elementnamen fungiert. Das Kommando ist so lange beginnend bei 0 mit steigenden Zahlenwerten aufzurufen, bis ein Fehler zurückgegeben wird.

CmdListUID <num>

Dieses Kommando liefert die UIDs aller in einem Projekt vorhandenen Elemente zurück. Es erwartet eine positive Zahl `num` als Parameter, welche als Zähler für die aufzulistenden UIDs fungiert. Das Kommando ist so lange beginnend bei 0 mit steigenden Zahlenwerten aufzurufen, bis ein Fehler zurückgegeben wird.

8 CorrCorrect

Dieses Kapitel beschreibt mit CorrCorrect ein Tool, welches es erlaubt, Korrekturtabellen für Scanköpfe und deren zugehörige Optik zu erzeugen, zu modifizieren und zu verfeinern.

8.1 Overview

Wenn ein Scankopf - ein System mit zwei Spiegeln, welche den Laser in X- und Y-Richtung ablenken - verwendet wird, so kann die sich ergebende Bewegung auf dem Material gegenüber den Ursprungsdaten verzerrt sein. Hier gibt es folgende mögliche Ursachen für Verzerrungen:

1. eine Kissenverzerrung verursacht durch die Spiegel und eine nachgeschaltete Optik
2. eine 3D-Fokusverzerrung welche durch die abweichende Distanz zwischen letztem Spiegel und Arbeitsbereich entsteht, weil der Laserstrahl in der Mitte des Arbeitsbereiches und an dessen Rand unterschiedlich lange Wege zurücklegt
3. eine räumliche Verzerrung durch einen Schrägeinbau des Scankopfes um z.B. innerhalb von Werkstücken zu arbeiten (bspw. die innere Seite eines Ringes)

Um solche Verzerrungen auszugleichen, kommen spezielle Korrekturtabellen zum Einsatz. In den meisten Fällen sind diese Tabellen Scankopf-spezifisch und werden auch mit diesen ausgeliefert. Diese liegen dabei in einem herstellerspezifischen Format vor, welches abhängig von der zu verwendenden Software bzw. dem Scannercontroller Plug-In unter Umständen noch konvertiert werden muss. In anderen Fällen existiert ein solches Korrekturfile gar nicht und muss von Grund auf erzeugt werden. Ein weiterer Anwendungsfall sind hochgenaue Applikationen, bei denen ein vorhandenes Korrekturfile nicht exakt genug ist und nachkalibriert werden muss, um genau zu einer vorhandenen Hardware zu passen.

All diese Aufgaben können mit dem Korrekturtabellentool "CorrCorrect" erledigt werden, welches hier beschrieben wird. Die sich ergebende Korrekturtabelle kann anschließend im BeamConstruct HD (.bco) Korrekturdateiformat gespeichert werden – einem speziellen Format mit besserer Genauigkeit und Auflösung als die meisten anderen Formate. Eine solche Korrekturdatei kann dann direkt mit einigen Scannercontroller-Plug-Ins des OpenAPC-Softwarepaketes verwendet werden (wie z.B. das ETH6608 Plug-In).

8.2 Benutzung

Um mit der Erzeugung einer neuen Korrekturtabelle zu beginnen kann entweder der "Neu"-Button in der Toolbar gedrückt oder der Menüpunkt "Projekt - Neu ausgewählt" werden. Diese Operation öffnet einen Dialog, in welchem Parameter für Fokus- und Kissenverzerrung gesetzt werden können. Zusätzlich kann festgelegt werden, welche dieser beiden Korrekturen tatsächlich angewendet werden soll. So ist es für reine 2D-Applikationen, bei denen gar keine dritte Achse existiert, nicht notwendig, eine Fokuskorrektur berechnen zu lassen. Hier genügt die Korrektur für die Kissenverzerrung. Für echte 3D-Applikationen kann auch die Fokuskorrektur verwendet werden.

Alternativ dazu ist es auch möglich, eine bereits existierende Korrekturdatei über die Toolbar (Button "Projekt öffnen") oder über den Menüpunkt "Projekt -> Öffnen" zu laden.

In einem nächsten Schritt kann die so erzeugte Basiskorrekturtabelle durch die Festlegung von ein oder mehreren Soll- und Istpositionen verändert werden. Um die dafür benötigten Daten zu erhalten, sind mit dem aktuellen Korrekturfile einige Geometrien oder einzelne Punkte zu markieren. Die Ecken dieser Geometrien bzw. diese Punkte sollten dabei an Positionen liegen, an denen die Genauigkeit verbessert werden soll. Alternativ können sie auch möglichst gleichmäßig über den Arbeitsbereich verteilt werden. Nach dem Markieren ist die tatsächliche Position dieser Punke zu messen. Diese Messwerte (=Istpositionen) müssen dann zusammen mit den Sollpositionen aus dem Markierprojekt in den Dialogen zur Korrektur einzelner und mehrerer Punkte eingegeben werden. Diese Dialoge sind über das Menü "Bearbeiten -> Einzelnen Punkt korrigieren" oder "Bearbeiten -> Mehrere Punkte korrigieren" zugänglich. Wird der Korrekturdialog mit "OK" verlassen, so wird die Korrekturtabelle an den Positionen der eingetragenen Punkte modifiziert um die gemessene Verzerrung auszugleichen. Dieser Vorgang muss so lange mit den gleichen oder mit anderen Positionen wiederholt werden, bis die sich ergebende Korrekturtabelle ausreichend genau ist.

ACHTUNG: Diese Art der Positionskorrektur beeinflusst die Korrekturtabelle nur in X- und Y-Richtung. Eine möglicherweise bereits enthaltene Fokuskorrektur wird durch diese Operation ungültig gemacht. Deswegen muss im Fall von 3D-Applikationen die Fokuskorrektur anschließend noch einmal erzeugt werden: durch Auswahl des Menüpunktes "Bearbeiten -> Fokuskorrektur neuberechnen" wird die 3D-Fokuskorrektur neu erzeugt ohne dabei die bereits vorhandene 2D-Korrektur zu beeinflussen.

Weiterhin beinhaltet CorrCorrect das gleiche "Geometrie"-Panel wie es bereits aus BeamConstruct bekannt ist. Wird eine Korrekturtabelle - die in der Mitte des Fensters angezeigt wird - selektiert, so können in diesem Panel verschiedene Operationen auf die gesamte Tabelle angewendet werden. Diese kann dort skaliert, gespiegelt, zentriert oder rotiert werden, es ist möglich ihre gesamte Position zu verändern und anderes mehr.

8.3 Korrekturdefinitionsdialog

Wird der Menüpunk "Projekt -> Neu" oder der Toolbar-Button “Neues Projekt” ausgewählt, so öffnet sich ein Dialog, in welchem sich die Fokus- und Kissenkorrekturparameter eingeben lassen. Ein separater Dialog welcher ausschließlich eine Fokuskorrektur ermöglicht, ist unter dem Menüpunkt "Bearbeiten -> Fokuskorrektur neuberechnen" zugänglich. Dessen Parameter sind eine Teilmenge des hier beschriebenen Dialoges und werden deswegen nicht separat erläutert. Der Dialog zur Definition von Korrekturparametern bietet folgende Optionen:

- Z-Abstand – der kürzeste Abstand zwischen dem Mittelpunkt des letzten Spiegels und dem Arbeitsbereich.

Für die Kissenkorrektur sind folgende Parameter wichtig:

- Checkbox Kissenkorrektur – die 2D- / Kissenverzerrungsparameter werden nur dann zum Erzeugen einer neuen Korrekturtabelle verwendet, wenn diese Checkbox aktiviert ist
- Winkel X – die Stärke der Verzerrung/Winkel in X-Richtung
- Winkel Y – die Stärke der Verzerrung/Winkel in Y-Richtung

Für die Fokuskorrektur kommen die folgenden Parameter zum Einsatz:

- Checkbox Fokuskorrektur – die Tiefen- / Fokusverzerrungsparameter werden nur dann zum Erzeugen einer neuen Korrekturtabelle verwendet, wenn diese Checkbox aktiviert ist
- Größe Arbeitsbereich – die Größe des Arbeitsbereiches beim gegebenen Z-Abstand; Unabhängig von der tatsächlichen Form wir immer ein quadratischer Arbeitsbereich angenommen, hier ist also die größte anzunehmende Kantenlänge einzutragen

8.4 Korrektur einzelner und mehrere Punkte

Die beiden Dialoge zur Eingabe von einzelnen sowie mehreren Korrekturpunkten erwarten die gleichen Werte und verhalten sich damit - mit Ausnahme der folgenden Eigenschaften - identisch:

- der Dialog zur Korrektur eines einzelnen Punktes erwartet Koordinatenpaare für exakt eine Soll- und eine Istposition
- der Dialog zur Korrektur mehrere Positionen erwartet eine CSV-Datei, welche Paare von Soll- und Istkoordinaten enthält, welche dann wiederum innerhalb des Dialogs als editierbare Liste angezeigt werden. Die CSV-Datei enthält Komma-separierte Werte im Format

  ```
  Soll-X, SollY, Ist-X, Ist-Y,
  ```

  Als Zahlenwerte werden Fließpunktzahlen mit einem Punkt als Trenner erwartet (also z.B. 97.651).

Darüber hinaus bieten beide Dialoge folgende Felder und Eingabemöglichkeiten an:

- Sollposition X, Sollposition Y – die Koordinaten der Sollposition in der Einheit mm
- Istposition X, Istposition Y – die Koordinaten der Istposition in der Einheit mm
- Arbeitsbereichposition X und Y – die Position der oberen linken Ecke des Arbeitsbereiches in der Einheit mm (wie auch in den Einstellungen des Scanner-Plug-Ins angegeben)
- Arbeitsbereichsgröße - die Größe des Arbeitsbereiches in mm, hier wird ein quadratischer Arbeitsbereich angenommen

Wird der jeweilige Dialog mit "OK" verlassen, so wird die aktuell geladene Korrekturtabelle entsprechend den vorgegebenen Soll- und Istpositionen modifiziert.

8.5 Dialog zur räumlichen Korrektur

Diese Art der Korrektur kann zum Einsatz kommen, wenn der Scankopf nicht parallel zum Arbeitsbereich angebracht ist, sondern schräg (um beispielsweise das Innere von Ringen zu markieren). Hierbei wird eine Korrektur als Basis benötigt, welche bereits zum verwendeten Scankopf und dessen Optik passen muss, anschließend kann dann die räumliche Korrektur als zusätzlicher Parameter hinzugefügt werden. Für einen korrekten Ausgleich der räumlichen Verzerrung müssen folgende Schritte ausgeführt werden:

- Montage des Scankopfes exakt parallel zum Arbeitsbereich, so dass der Abstand zwischen Kopf und Arbeitsbereich genau so groß ist, wie der kürzeste Abstand zwischen Kopf und Arbeitsbereich bei der späteren schrägen Montage
- Überprüfen, ob die Korrektur in dieser Einbaulage für den Scankopf korrekt ist; sollte diese nicht stimmen, ist sie mit den oben beschriebenen Funktionen anzupassen
- sobald sich mit den Korrekturwerten ein exaktes Quadrat markieren lässt, ist dessen Größe zu messen und der Kopf in der gewünschten Schräglage zu montieren
- jetzt ist das Quadrat erneut zu markieren, wobei dieses verzerrt dargestellt wird, bei diesem ist die Kantenlänge an der Seite, an der der Abstand zwischen Arbeitsbereich und Scankopf am größten ist, erneut zu messen

Die sich daraus ergebenden zwei Werte - die normale (erwartete) Kantenlänge des Quadrates, welche im ersten Schritt gemessen wurde und die zweite, größere Kantenlänge - sind im Dialog zur räumlichen Korrektur einzutragen. Als zusätzliche Information muss über die Radiobuttons noch die Seite der Korrektur markiert werden, an der die gestreckte Kante gemessen wurde. Nach Betätigung des "OK"-Buttons wird die räumliche Korrektur in die aktuelle Korrekturtabelle einberechnet, so dass die Tiefenverzerrung wärhend der Markierung aufgehoben wird. Das Markierergebnis ist jetzt ein Rechteck, da das Quadrat noch entlang der Neigungsrichtung gestreckt ist. Um dessen Länge zu korrigieren kann entweder der Gain-Faktor in den Einstellungen des Scanner-Controller-Plug-Ins (in BeamConstruct) angepasst werden, oder die Korrekturtabelle als ganzes im Gemeotrie-Panel von CorrCorrect in der enstprechenden Richtung skaliert werden.

Index

1

11350 ... 41

A

Allgemeine Stifteinstellungen ... 37
An Raster ausrichten ... 35
Anweisungsliste ... 9, 24
Applikation beenden ... 36
Arbeitsbereich ... 32, 35
AWL ... 9

B

BeamConstruct ... 6, 30
BeamLock ... 40
BEAMP ... 33
BeamServer ... 41, 42
Benutzer ... 35
Benutzer verwalten ... 35
Benutzerrechteverwaltung aktivieren ... 35
BINÄR ... 14

C

CmdAppendPrj ... 45
CmdListName ... 47
CmdListUID ... 47
CmdLoadPrj ... 45
CmdNewPrj ... 45
CmdRedraw ... 47
CmdRefresh ... 47
CmdSendMark ... 46
CmdSendNamedMark ... 46
CmdSetCharIO2 ... 46
CmdSetCharIO3 ... 46
CmdSetCharIO4 ... 46
CmdSetCharIO5 ... 46
CmdSetSelEnt ... 46
CmdStopMark ... 44
CmdWriteMark ... 46
CmdZoomElem ... 47
CNC ... 14
CNConstruct ... 6, 28
ControlRoom ... 6, 7

D

Debugger ... 7
DIGITAL ... 13

E

E1701D ... 16
E1701M ... 19
E1701M Schrittmotor-Controller ... 19
Einstellungen verändern ... 36
Elementtree in Tabbed Pane verschieben ... 35
Ergebnis verfeinern ... 37
ExitUI ... 41, 43

F

Flow-Editor ... 11, 12, 14
Flow-Objekt ... 13
Frequenz ... 33

G

G-Code ... 14, 15
G-Code Controller ... 14
G00 ... 15
Geometrie bearbeiten ... 36

H

HALaser E1701D Scannercontroller ... 16
Hatch ... 33
Hintergrundfarbe ... 35
HMI ... 6
HMI-Editor ... 11, 12

I

IL ... 24
Instruction List ... 9, 24
Interlock Server ... 7, 22, 25
Interlock Servers ... 24, 25
InterlockServer ... 6
IOSelect-Modus ... 36, 38

L

Laserleistung ... 33
Laserparameter ... 33
liboapc ... 24
LUA ... 9, 24

M

Markieren ... 34, 36
Markiergeschwindigkeit ... 33
Multihead ... 37

N

Not-Aus ... 34
Notstopp ... 34
NUMERISCH ... 13

O

Oberaufsicht ... 35
OpenDebugger ... 6, 7, 22
OpenEditor ... 6, 7, 11
OpenHPlayer ... 6, 7, 24
OpenIServer ... 7, 24, 25
OpenPlayer ... 6, 7, 23
OpenPlugger ... 6, 7, 27
OpenPluggers ... 27

P

Parameter für Stift verwenden ... 37
Parameterwizard ... 37
Plugged Devices ... 11, 22
Projekt ... 35
Projekt laden ... 36
Projekt speichern ... 36
Projekteinstellungen ... 35
Prozesssteuerung ... 6

R

Raster ... 35
Remote Control Kommandos ... 43
Rezept laden ... 36

S

Scanner ... 32
Scanner-Delays ... 33
Scannercontroller ... 32
Scannerkarte ... 32
Schraffurmuster ... 33
Sequencer ... 9
Sprunggeschwindigkeit ... 33
Stifteinstellungen ... 36
Stiftparameter-Wizard ... 37

T

Tastaturabkürzungen ... 35
TriggerUI ... 44

V

Verschiedenes ... 36
Von diesem Ergebnis zurückgehen ... 37

Z

ZEICHEN ... 14
Zeichenbereich ... 35